2002年4月1日発行（毎月1回1日発行）第21巻第4号 昭和57年12月6日第三種郵便物認可

マイコン

わかる！動かせる！
プログラムが組める雑誌

BASIC Magazine

APR. 2002
4

定価
550Yen



掲載プログラムがダウンロードできる！ http://www.basicmagazine.net/

新連載 見やすい楽しい誰でも使えるサイトを実現！

一歩先ゆくHP（ホームページ）作り

ユニバーサルデザイン導入で訪問客を増やす！

BM基礎講座
ベーシックプログラミング教室
第1回 プログラミングの第一歩

プロセッサGHz時代到来！
CPUの歴史と未来をさぐる！

ハードディスクは忘れない！
マンガで学ぶ！コンピュータの世界

実践！8Mbps ADSL回線の接続方法
これからはじめるブロードバンド活用法

特選投稿プログラム・コーナー

Visual Basic V6.0	スピード・スター
Visual Basic V6.0	Strike Number
Visual Basic V6.0	L-R Typing
Visual Basic V4.0	BALL & ARROWS
Visual C++	Dust
Visual C++	3D Shoot
Delphi	ぶろっくランド
F-BASIC	チェッカー2
HSP	物体捕獲
iアプリ	アミダアミーゴ
iアプリ	ぷにょんぷにょん
WonderWitch	月と太陽



三鷹市立第一小学校
FOMAを使った体験授業公開！





編集部の学校訪問記
実践的教育で資格取得、
プログラミング・コンテスト常連の伝統校
千葉県立一宮商業高等学校

教科「情報」など授業にも使えるプログラム満載◀

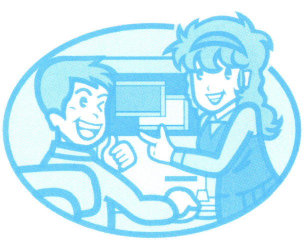
マイコンBASIC Magazine

http://www.basicmagazine.net

4 2002 APRIL







© マイコンBASICマガジン4月号
第21巻 第4号 2002年3月25日印刷
2002年4月1日発行
定価 550円（本体 524円）
電波新聞社 2002 Printed in Japan
編集発行人：平山哲雄
東京都品川区東五反田1-11-15
郵便番号 141-8715
電話　東京(03)3445-6111(大代表)
振替（東京）00150-3-51961
印刷：奥村印刷(株)　製本：(株)堅省堂

BASIC Magazine APRIL 2002 CONTENTS

・5月号は4月8日発売です！・

広告索引

マイコンソフト

XVGA-1PROとは

XVGA-1PROは、DOS/Vパソコン（AT互換機）をはじめ、NEC PC-9800シリーズ、Apple PowerMacintoshの画面出力を、家庭用テレビで見ることができるように映像信号を変換する装置です。
大型テレビや、液晶プロジェクターなどに接続して、パソコンを利用したプレゼンテーションをする際に適しています。
本機は、多くの映像端子出力をそなえており、ビデオ端子（※）、Sビデオ端子（※）、HD/コンポーネント・ビデオ端子（※）、D端子（※）を持つ家庭用テレビや映像機器に接続し、パソコン映像を表示することが出来ます。
本機で変換された映像は、NTSCだけではなく、ハイビジョンやVGA（水平同期周波数31kHz）の映像としても出力できるため、お客様がハイビジョン対応テレビや、パソコン映像入力対応テレビをお持ちの場合は、NTSCより鮮明な映像を表示することができます。
（※）各端子の形状はカタログ裏面をご覧下さい。



高画質TVは、より鮮明な映像出力コンバーターを選ぶ
ニュータイプD端子／HD端子対応ダウンスキャンコンバーター登場！
こんなコンバーターが欲しかった!!

オン・スクリーン・メニューとリモコン操作も充実

変換した映像の表示位置や明るさなどは、テレビ画面に表示された項目を選択して調整できます。また、離れた位置から操作ができる『リモコン』やXVGA-1 PRO本体の設置場所によってリモコン受光部が隠れてしまっても、リモコンを使用できるように『リモコン受光ユニット』も付属されています。

使って便利な画面ズーム機能

画面の拡大機能を持つスキャンコンバーターは、珍しくありませんが、XVGA-1PROの拡大機能は、より使いやすくなるように工夫されています。
9等分されたテレビ画面がリモコンに割り当てられ、拡大したい位置を直感的に選ぶことができる『エリアズーム機能』。リモコン上の3つのボタンに拡大したい位置と拡大率を登録できる『ホーム位置登録機能』などがあります。
拡大した画面は、拡大率を変更したり、上，下，左，右，斜め，などの方向に表示位置を移動させることができます。

ワイドテレビの画面比率もバッチリ！

4：3のテレビと16：9のワイドテレビでは画面比率は異なります。通常のテレビ放送もスキャンコンバーターで変換した映像もワイドテレビで表示すると、横長になってしまいますがXVGA-1PROではこれらの問題を解決するため、『WIDE_TV』機能が付いています。この機能を使用すると、ワイドテレビでも画面比率1：1の映像を表示することができます。

新しいパソコンからPC-9801まで対応！

最近のDOS/V（AT互換機）パソコンやApple Power Macintoshはもちろん、PC-9801シリーズの24kHz（640×400ドット）にも対応しています。
24kHzのRGB信号をNTSC信号に変換する機能は、他のコンバーターでもできますが、本機は24kHz（640×400ドット）を31kHz（640×480ドット）にアップスキャンすることができます。これにより、入手が困難になった24kHz対応モニタの変わりに現在のDOS/V用モニタが代用できることになります。また、XGA（1024×768ドット）やSVGA（800×600ドット）の画面をVGA（640×480ドット相当）に変換して表示する機能を備えていますので、VGA画面（640×480ドット）しか表示できないパソコン対応テレビやVGAプロジェクターにもXGA画面やSVGA画面を表示することができます。

対応機種と対応画面モード

対応機種	対応画面モード	水平同期（垂直同期）
NEC PC-9801シリーズ	640×400ドット	24kHz（56Hz）
各社DOS/Vパソコン（AT互換機）または NEC PC-9801/9821 NEC PC-98NX	640×400ドット	31kHz（70Hz）
	640×480 ドット	31kHz（60Hz）
		37kHz（72Hz）
	800×600ドット	35kHz（56Hz）
		37kHz（60Hz）
		48kHz（72Hz）
		46kHz（75Hz）
	1024×768ドット	48kHz（60Hz）
		56kHz（70Hz）
		60kHz（75Hz）
Apple Power Macintosh ※4	640×480ドット	35kHz（67Hz）
	800×600ドット	37kHz（60Hz）
	832×624ドット	49kHz（75Hz）
	1024×768ドット	48kHz（60Hz）
		61kHz（75Hz）

※4 Macintoshは Apple Computer社の登録商標です。

ハイビジョン・テレビ対応ビデオ・スキャンコンバーター・ユニット

好評発売中

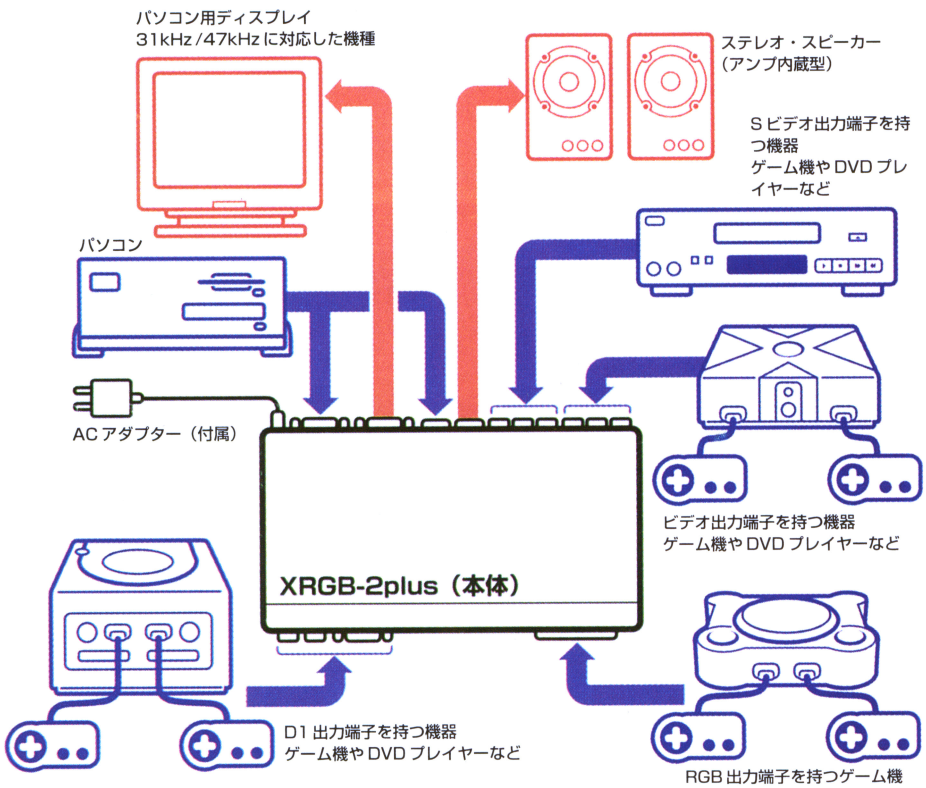
XVGA-1PRO
VIDEO SCAN CONVERTER UNIT

標準価格56,800円（税別）
JANコード 4961997001066 DP3913360

マイコンソフト

マイコンソフト社ホームページ
http://www.micomsoft.co.jp/index2.html

新発売





アップ・スキャンコンバーター・ユニット

XRGB-2plus

標準価格**28,800**円（税別）　DP3913384000　JANコード4961997001073

XRGB2-plusの特徴

XRGB-2plusは、ご使用のパソコン用ディスプレイに、家庭用ゲーム機やDVDプレイヤーの映像を表示するための映像変換装置です。通常パソコン専用ディスプレイは、端子の違いや対応周波数が違うため、家庭用のゲーム機やDVDプレイヤーの映像を表示することが出来ませんが、XRGB-2plusを使用すれば、これらの画面を表示することが可能になります。パソコン用ディスプレイにパソコン映像を表示するだけでなく、ゲーム用ディスプレイなどとしても利用できるのです。XRGB-2plusは水平同期周波数31kHz（640×480ドット）の画面モードと、水平同期周波数47kHz（1024×768ドット）の画面モードを持つパソコン専用ディスプレイに接続し、画面を表示することができます。一台持っておけば、何かと便利な装置です。

■仕様

映像方式	NTSCカラー／アナログRGB
入力	コンポジット・ビデオ（NTSC準拠） 1Vp-p（75Ω） S端子（NTSC準拠）4ピンmini DIN Y信号 1Vp-p C信号 バースト信号部で0.286Vp-p（75Ω） D1端子（NTSC準拠・525i） 14ピン・アンフェノールハーフピッチ Y信号 1Vp-p PB信号 0.7Vp-p（±350mVp-p） PR信号 0.7Vp-p（±350mVp-p） 21ピンRGBマルチ端子（アナログRGB） RGB信号 0.7Vp-p 複合同期信号 0.3Vp-p 15ピン・アナログRGB端子（スルー出力用） HD-SUB
出力	15ピン・アナログRGB端子 HD-SUB 電源オン時：RGB信号 0.8Vp-p（75Ω） 同期信号 TTLレベル （V：60Hz、H：31kHz/47kHz） ※出力同期信号は入力される映像信号によって若干異なります。
画質	電源オフ／スタンバイ時：15ピン・アナログRGB入力端子（PC IN）からのスルー出力 電源オン時：ビデオ音声入力、Sビデオ音声入力、D1音声入力、21ピンRGBマルチ端子の各端子より音声を入力。 電源オフ／スタンバイ時：パソコン音声入力端子から音声を入力。 電源オン時：ビデオ音声入力、Sビデオ音声入力、D音声入力、21ピンRGBマルチ入力の各端子から音声を出力。 電源オフ／スタンバイ時：パソコン音声入力端子からの音声をスルー出力。 水平同期周波数31kHz時 （サンプリング周波数14MHz） 水平同期周波数47kHz時 （サンプリング周波数14MHz） 色階調：R=8bit、G=8bit、B=8bit トータル1,600万色（フルカラー）
Y／C分離	3ライン・デジタルくし型フィルター
電源	専用ACアダプターにより供給 DC 12V 最大消費電流 1A　最大消費電力12W
本体寸法	W230mm×D130mm×H39.25mm （ボタン、コネクター、ゴム足を除く）
重量	本体：約480g　電源アダプター：約132g

※重量は生産時期によって若干変わることがあります。

発売元：電波新聞社　販売部
〒141-8735 東京都品川区東五反田1-11-15　☎03-3445-8201（ダイヤルイン）

企画・開発：マイコンソフト株式会社
〒530-0005 大阪市北区中之島3-2-4 朝日新聞ビル6F（電波新聞社大阪本社内）☎06-6203-2827

東京本社 03-3445-6111（大代表）	札幌支局 011-641-5591	関東総局 0273-26-3206	名古屋支局 052-261-4541	広島支局 082-228-5581
大阪本社 06-6203-3361（大代表）	仙台支局 022-227-7211	松本支局 0263-36-2266	金沢支局 076-263-8661	高松支局 0878-61-3111
西部本社 092-431-7411（大代表）	新潟支局 025-245-2526		京都支局 075-221-8021	鹿児島支局 099-226-3630



編集部の学校訪問記

千葉県外房，九十九里の南端に位置し，海水浴やサーフィンのメッカとしても知られる一宮町。無形文化財指定の「はだか祭り」でお馴染みの玉前神社のすぐそばに一宮商業高校（安藤政之校長）はあります。前身は大正14年設立の私立一宮実業学校，昭和28年県立一宮商業高等学校として独立，専門的で実践的な教育が実践されている伝統校です。同校は全国に先駆けて情報処理教育を推進してきており，資格検定に合格できるレベルの授業内容，進学先や就職先で十分に対応できるだけの技能修得を目標にしています。

今回は，「情報処理科」の授業のようすと，プログラミング・コンテスト最優秀賞の常連「電算部」を取材してきました。

千葉県立 一宮商業高等学校

http://academic1.plala.or.jp/ichisyo/

実践的教育で資格取得，プログラム・コンテスト常連を目指す伝統校

全国でもいち早く情報処理教育に取り組み，実績を上げている

同校には現在，商業科と情報処理科が設置されています。情報処理教育にいち早く取り組みはじめ，昭和46年，全国で3番目に「情報処理科」が設置されました。

情報処理教育は，1年，2年は資格検定合格を目指す授業内容で，生徒が自分でコースを選択できるようになっています。3年はできるだけ多くのアプリケーションソフトを使いこなし，進学先，就職先で十分使いこなせるための基本知識と技能修得を目標にしています。

在学中に受験できる資格検定試験には，コンピュータ利用技術検定／基本情報処理技術者試験／初級システムアドミニストレータ試験などがあり，多くの合格者を出しています。

【情報処理科】

1年

情報処理（2単位）

経産省第2種情報処理技術者試験のハードウエアの内容と全商情報処理検定の1級から3級の用語の内容

プログラミング（4単位）

10月まで全商情報処理の2級の範囲を終了し，その後1級コースと2級復習コースに分かれ（約7割の生徒が1級チャレンジ）授業を展開，1月の検定では全員合格を目指す。

2年

プログラミング（3単位），経営情報（2単位）

2年次は4月当初から，2種コース，システムアドミニストレータ（シスアド）＋利用技術検定コース，情報処理１級＋利用技術検定コースの3コース制をとり，4人で担当。どのコースにするかは生徒の自由選択。

3年

総合実践（4単位）

- 応対のしかた，基本取引
- Excelを使った表とグラフ処理
- Wordを使ったワープロ処理
- Accessを使ったデータベース［販売管理システムの作成］
- PowerPointによるプレゼンテーション資料の作成
- インターネットの活用
- イントラネット上でのホームページの作成と参照，電子メールの送受信

経営情報（2単位）

Visual Basicを使ったプログラム作成。平成9年度から取り組んでいる。

【商業科】

2年

情報処理（3単位）

- 表計算（Excel）コンピュータ利用技術試験2級取得を目指す。
- データベースの利用（Access）データの検索，簡易なデータベースの作成
- インターネットwww, e-mail, ホームページの作成

3年

プログラミング（3単位）選択

- 表計算（Excel）コンピュータ利用技術試験の1級取得を目指す
- Visual Basicの基本

電算棟には日立のFLORAが133台

パソコンを使用した実習は「電算棟」で行われます。

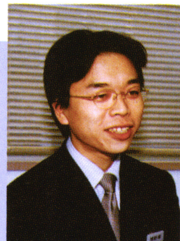
1階にはWindows2000マシンが43台。Visual BasicやMS Office系のアプリケーションを中心に使用されています。2階はWindows2000マシンが45台。主に情報科がCOBOLやHP制作で使用しています。3階はWindows Meマシンが45台，主に商業科の授業で使われています。133台すべて日立FLORAが導入されています。

授業の他，生徒たちは放課後も使用可能で，各自の課題制作や検定試験対策などで活用しているそうです。



▲安藤政之校長先生。情報処理教育の良き理解者





▲情報処理科の科長・渡部　清先生（左）と鵜野澤 博先生（右）。お2人とも電算部の顧問でもある

情処科3年生の授業



◀2学期の終わり頃から取り組んだというホームページ制作。イントラネット上でお互いを評価する

▼Visual Basicで課題研究の仕上げ中。スロットマシンなどのゲームが例題として出題され，さくさく作成







「プログラミング部門」「コンテンツ部門」両方最優秀賞をとろう！を合言葉に活動する電算部

情報処理科・商業科の生徒それぞれ半々，現在26名の部員で構成されている「電算部」。プログラムが作れ，ライセンスも取れるという2本柱で，授業で十分に行えないプログラム作成能力を高めるため，Visual Basicを使い，CAIソフト中心の制作に日々励んでいます。

その歴史は古く，昭和50年代から各コンテストに毎年参加，全国商業高等学校協会主催のプログラミング・コンテストでは，過去10年で9度「最優秀賞」を受賞している常連校です。さらに，経済産業省主催プログラミング・コンテストのコンテンツ部門で平成11年と12年の2年連続「優秀賞」を受賞，13年度にははじめて最優秀賞を受賞しました。

また，情報処理検定，コンピュータ利用技術検定，ワープロ検定など検定にも積極的に取り組んでいます。

1年
- 4月～6月…ワープロ・ペイント・VBの基礎
- 7月～8月…プロコン作品の手伝い（画像作成）
- 9月～11月…文化祭出品のための作品制作
- 12月～1月…資格取得
- 2月～ 3月…flashやshadeなどのソフトの活用・研究／プロコン作品のテーマ検討

2年
- 4月～8月…プロジェクト（5～7名）を作り，作品制作に入る（全商協会へ出品）／夏休み2度の合宿
- 9月～ …資格取得や自由研究

3年
- 2年次と同様に，作品制作（経済産業省へ出品）

経済産業省主催 全国高校生プログラミング・コンテスト コンテンツ部門

最優秀賞受賞「もじもじかずかず」



3年 矢澤浩一くん／藍佑次くん／西岡由貴さん／石井智子さん／田中優子さん／石川美帆さん／露崎俊之くん

幼年期の子供たちが文字と数を学習するためのWeb上で動作するソフト。教育テレビがヒントになったというこの作品は，クイズ感覚で学習でき，音声や豊富なイラストと動画が使用されている。







全国商業高等学校協会主催 プログラミング・コンテスト

最優秀賞受賞「日本の心～百人一首」

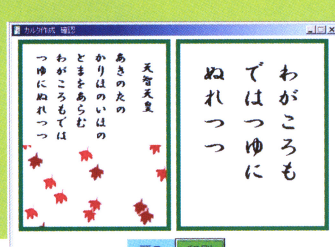




2年 田辺裕美子さん／湯浅将也くん／永嶋のり子さん／米本美央さん／今川　希さん

古典が好きということで，取り組み始めた作品。小学校低学年レベルで，百人一首をいろんな角度から学ぶ。①単語の意味を覚える②ゲーム感覚でプレイ③オリジナル一首を作るなど





優秀賞受賞「応急こどもガイド」



2年 田中三由希さん／木嶋朋美さん／緑川あやのさん／石渡絵美さん／中村有里子さん／堀江智子さん



小さな子供のいる若いお母さんを意識して作ったという作品。こどもが怪我などをしたときの処置をゲーム感覚で確認していくもの。インターネットや本で調べ，それをイラストなどでわかりやすく表現している。



2000年7月に公開されたホームページ。立ち上げ当初から電算部の生徒も参画している

一宮商業高等学校　〒299-4301 千葉県長生郡一宮町一宮3287　☎0475-42-4520

学校訪問ルポ

三鷹第一小学校でFOMAを使った実験授業を公開



◀FOMAを使ってインターネットライブ放送



◀FOMAのテレビ電話機能を通じて、指導をあおぐ児童たち



▶自分たちで考えたルールをもとにソフトバレーボール



◀指導する末永先生

本誌2002年1月号の学校訪問記でも紹介した東京都の三鷹市立第一小学校（松田洋子校長）で2月5日，FOMAを使用した体育の公開授業が行われました。

5年3組担任の末永寿宣先生が，東京都教育研究員（教育課題）として一年間取り組んできた研究成果の一端を報告するためのもので，研究主題は「自立的に行動する子供を育てる指導法の工夫～体育・学級活動の指導を通して～」。

「ソフトバレーボール」のルールをみんなで考えながら実践することで，自立的な規範行動がとれる子供を育てようというのがねらい。

子供たちが集まった体育館には，FOMAを3台用意。1台はインターネットライブ放送用で，同校のホームページ上で公開授業の様子を流すためのカメラとして使用。2台は，自宅で待機する日本バレーボール協会公認ソフトバレーボールリーダー・西野入瑛子さん，三鷹市体育指導委員協議会委員イントラネット地域協力者・高麗美智子さんとの連絡用。FOMAのテレビ電話機能を通じて，サーブ権や得点のルールについて相談し決定，子供たちに指導していました。

末永先生は「専門家の意見を聞くには，大がかりなセッティングが必要だった。FOMAによって，遠隔地からでも適切な指導が可能になり，子供たちも楽しんで取り組んでいた」と，その効果の大きさを力説しています。

FOMAを導入しているのは，すべて三鷹市の学校で，全国でも5校だけ。NTTドコモの城石　勇SEは「FOMAはこれ自体で簡単なデジタルビデオカメラにもなる。今後は野外学習で先生との連絡用やWeb上にあるポケット電子図鑑と実物を見比べるなど，活用用途を広げていってもらいたい」と話していました。

三鷹市立第一小学校
〒181-0004
東京都三鷹市新川6丁目4番32号
☎0422-43-1177

文部科学省の先進的教育ネットワーク地域指定校として，1997年から校内ネットワークを導入，現在では無線LANやノートPC，PDAを利用した遠隔授業を取り入れていれており，「児童のいる場所，行く場所，どこにでもパソコンがやってくる」を実践している。

また，家庭・地域とをインターネットで結んだイントラネット構築の実験を進めている。

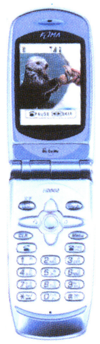
次世代移動通信サービス「FOMA」

http://foma.nttdocomo.co.jp

「FOMA（フォーマ）」はNTTドコモの次世代移動通信サービス。FOMAは最新の「W-CDMA方式」を採用することで，クリアな音声通話，高速なデータ通信，そしてTV電話機能などを実現しています。

通信速度はiモード（パケット通信）時で今までの約40倍に向上しています（下り最大384kb/s，上り最大64kb/s）。通話をしながらｉモードを利用するといった「マルチアクセス」機能にも対応しています。

「ビジュアルタイプ」では，会話しながら相手の表情も読みとれる「TV電話」機能が利用できます（64Kデータ通信を適用）。「スタンダードタイプ」では，iモードで映像や音を取得して楽しむ「iモーション」というサービスに対応しています。映像や音楽などのコンテンツを配信する情報端末としてもFOMAは活躍していきそうです。



▶iモーションに対応したスタンダードタイプ「FOMA N2002」



▶高速データ通信、クリアな音声通話を実現したスタンダードタイプ「FOMA N2001」



▶TV電話が楽しめるビジュアルタイプ「FOMA P2101V」



▶iモーション対応ビジュアルタイプ「FOMA D2101V」



▲PCカード一体型のデータタイプ「FOMA P2401」

FOMAは昨年10月のサービス開始以来，端末が高価なことや，通話エリアが限られていること（東京都心周辺，大阪・京都・名古屋など）もあり，まだ十分な契約者数には達していません。しかし今後，新機種の発売や通話エリアの拡大で，シェアを伸ばしていくものと思われます。

三鷹市立第一小学校URL→http://www.education.ne.jp/mitaka/ichisho-es/index.htm

見やすい，楽しい，誰でも使えるサイトを実現！

一歩先ゆく ホームページ作り

第1回 ユニバーサルデザイン導入で訪問客を増やす！

白木ミホ

作成支援ソフト花盛りの今，ホームページは誰にでも簡単に作れるものになりました。でも「なんとなくできちゃった」の一歩先ゆきたいキミなら，タグなどの基本をしっかりおさえて小技も効かせつつ，＜ポリシー＞のあるホームページを作ってみませんか？

そのために提案するのが「ユニバーサルデザイン」です！

ユニバーサルデザイン ＝使い勝手のいい形

今，「ユニバーサルデザイン」という言葉を知っていたらキミはかなりの情報通ですが，知らなくても大丈夫。それは，みんなが日常生活の中で何となく感じている「使い勝手のいいデザインとは？」という疑問につながっているからです。

例えば目の不自由な人でも問題なく使える自動販売機，お年寄りにも扱いやすい家電，あるいは利き手の左右を問わず使える道具などは，すべて「誰にでも使いやすいユニバーサルデザインの道具」と言えます。なんだ，特別な人のためのデザインか，というのは甘い！　キミは部活の帰りに筋肉痛で駅の階段を上がれず，エレベータを使ったことはないですか？　パソコンショップで買い物しすぎた帰り道，両手がふさがった状態で自動改札機を通ろうとしてエライ目にあったことはないですか？　今の季節に花粉症で涙目になって淡い色の文字が読めない人はいませんか？

そんな時に思いますよね？　もうちょっと考えて作ってくれよな～，と。

それがユニバーサルデザインの原点なのです。



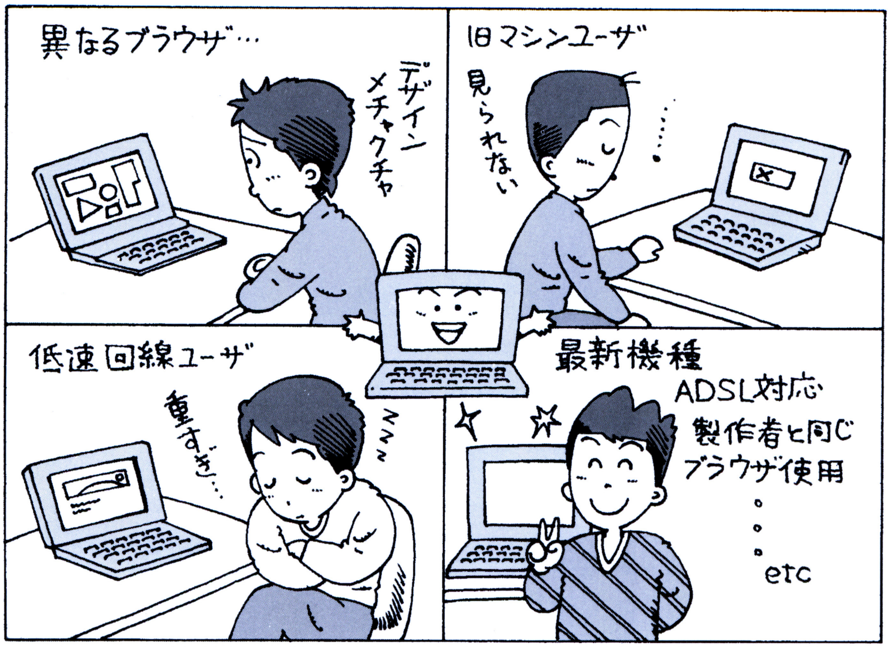
《第1図》生活の中のユニバーサルデザイン：自動改札機の場合「やさしいデザイン」は特別な人のためと思われがちだけど，右下のような例もありますね

ホームページのユニバーサルデザイン

インターネットのホームページの世界にも，もちろん「考えて作ってほしいこと」はたくさんあります。

ネットサーフィンをしているキミなら，こんな経験はありませんか？

いつもネットスケープで見ていたホームページをたまたまインターネットエクスプローラで見たら，あまりの違いにガクゼンとした！　とか，わくわくお楽しみでアクセスしたホームページが何だか凝った仕掛けを使ってて，しかも自分のパソコンの旧ブラウザでは見られなかった！　とか，はたまた画像系ホームページにアクセスしたら容量が大きすぎて，いつまでたっても画面が現れずがっかり



《第２図》ホームページのユニバーサルデザイン。数々の条件をクリアしないと快適に見られないのでは，良いホームページと言えません

した！　とか。

でも，いくら腹がたったからといってモニタにパンチを食らわすのは筋違い。罪はあなたのパソコンにではなく，数々の条件をクリアしなければ快適に見られないホームページ側にこそあるのです。

そして，それをクリアするのがユニバーサルデザインの使命なのです。

ホームページを耳で読め！

障害を持つ人たちにとって情報障害はとりわけ深刻な問題です。

世の中にはホームページを耳で読んでいる人が大勢います。目の不自由な人たちはホームページの情報を音声出力装置（音声リーダ）で「聞いて」いるのです（**第３図**）。

こうしたまったく異なるブラウザを使用する人にとって，目で見ることだけを特化したホームページのデザインは使えません。色の識別の難しい人，小さな文字の見えづらい人にとっても一部のデザインは問題となります。さらにインターフェースの問題として，マウスが使えない肢体不自由の人や，逆にキーボードの使えない人にとって，それなしにはアクセスできないホームページは「使えない」と言えるでしょう。



《第３図》ホームページを耳で読む。第５図の画面を音声リーダに通すと，こんなふうに聞こえます

自宅にいながらにして世界中から情報を集められるインターネットの存在は，障害者の情報環境を変える画期的な技術，と期待されていたにもかかわらず，現実は違ってしまっています。バリアフリーという言葉も最近よく聞きますが，インターネットの世界もいまだにバリア＝支障だらけなのです。

そこでユニバーサルデザイン先進国のアメリカでは，政府や公共機関のホームページに，それらの支障を取り払ったユニバーサルデザインを義務づける法律が誕生し，企業の公式ホームページなどもそれに従っています。

日本でもＵＤＩＴ（ユーディット）という団体がWebのユーザビリティー（使い勝手の良さ）やアクセシビリティー（利用しやすさ）を向上させるためのガイドラインを発信するホームページを通じて活動をしています（**写真１**）。

とはいえ，これらの「使い勝手」は何も特別な人のためのデザインには限りません。誰にでも利用しやすいということは，前述したようにどんなブラウザ，どんなマシン環境，あるいはどんな回線であっても快適にホームページを見られるようにしよう，ということだからです。

反面，最近のホームページには「動作環境：IE5.0，画面サイズ1024×768，フォント小推奨」などといった文をよく



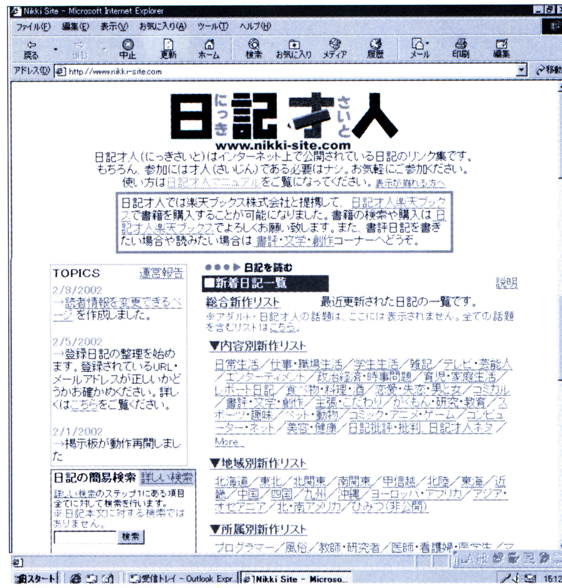
《写真１》ユーディットのホームページ。ユニバーサルデザインに基づいた，使い勝手の良いホームページのガイドラインやチェックリストなどがあります　http://www.udit-jp.com/



《写真２》ＩＢＭの『バリアフリーの扉』。ご存知ＩＢＭの公式サイト内でもユニバーサルデザインは紹介されています。音声リーダ「ホームページリーダー」の試用版がここからダウンロード可能　http://www.ibm.com/jp/



《写真3》ねえさんたちに100の質問。読むだけでも面白いけど，ぜひ参加したい自己紹介ワールド
http://www5d.biglobe.ne.jp/~kakigai/neisan/



《写真４》日記才人。日記専門の驚異の検索サイト
http://www.nikki-site.com/

見かけます。これは使い勝手という考え方とは正反対の，困った流行でしょう。敷居の高いレストランのように客を選んでいるわけです。何より環境を限定して楽になるのは制作者だけだからです。

ですが，みんながアクセスできるユニバーサルデザインのホームページは，ちょっと知識をつけさえすれば作ることができます。そんな「できる」「技アリ」のホームページを作りましょう！

ホームページで何を作るか？

ユニバーサルデザインの話からはちょっとそれますが，ホームページ制作ガイドである本連載にとって重要なことをもう一つ触れておきましょう。それは「ホームページといっても何を書いたらいいのか分からない」という，内容に関する最大の？疑問です。

自分のホームページを立ち上げたいという人の間で意外と深刻な？問題は実はテーマです。マニアやオタクな人なら即決でしょうが，大部分の人にとってははじめにホームページという道具があって，それを使って何を作ろうか，と考えるのがほとんどなのだとか。

一方で世の中を見回せば，ホームページとは個人が世界に向けて情報を発信できる最高の媒体である，なんて言われていて，ならば世界にふさわしい広大なテーマをご立派に書かねばならんのかとビビらされることもありますよね。

実際，ホームページの世界は今や「発信する」だけでは芸がありません。

しかし，本を発行するのと違って，ホームページは始めから完成品である必要がないのが大きな利点です。初日は表紙だけ作って，とりあえず世に出してしまう。それから１ページずつ書き足していく。まさか本でこれをやるわけにはいきませんが，ホームページはむしろその方が読者にとって都合がいい，というのも突然500ページに及ぶ大河ホームページが出現されても読みきれないからです。そう考えるとホームページの作成は案外お気楽なものなのです。

自己紹介と日記で燃える！

さて，気楽になったところで肝心の中身ですが，いちばん手軽に誰もが書ける「自己紹介」が今けっこうアツい！　というと意外でしょうか。

最近ブームを呼んでいるのが，100の質問に答えると，それがそのまま自己紹介になる『ねえさんたちに100の質問。』（**写真３**）と，その形式を継承してジャンルを特化した，いわゆる「100Ｑ（ひゃっきゅー）もの」です。答える人も読む人も，さらには自分で新規に「100Ｑ」を立ち上げる人も大いに楽しめて，リンク集も賑わっています。

さらに簡単に毎日内容を増やしていけることから，これまたお手軽ジャンルと思われがちなのは「日記」ですが，これこそは現在の日本のホームページ界にあって最もディープで最も熾烈なジャンルと言えます。

日記専門の検索サイト『日記才人（にっきさいと）』（**写真４**）には，１万以上もの日記が登録され，１日で200以上のアクセスを稼ぐような日記がひしめいています。特に評価の高いものは，文だけで読ませる・楽しませる「テキストサイト」と呼ばれるホームページで，もはや下克上・実力主義の戦国時代の様相です。まあ始めからそこまで思いつめる必要もないとは思いますが，上級者をも飽きさせないジャンルであるというのはいいことです。

また，家族単位でホームページを立ち上げる本当の「ホーム」ページも花盛りです。学校のクラス単位で作る「ホームルーム」ページもまとめあげれば，貴重な財産になるでしょう。

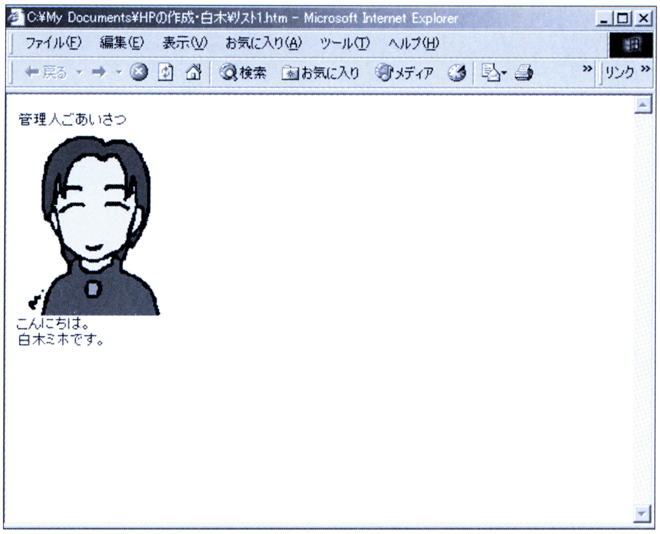
もちろん，絵や写真やエッセイといった自分の作品やコレクションの展示場としてもホームページはうってつけです。

いちばん簡単なホームページ

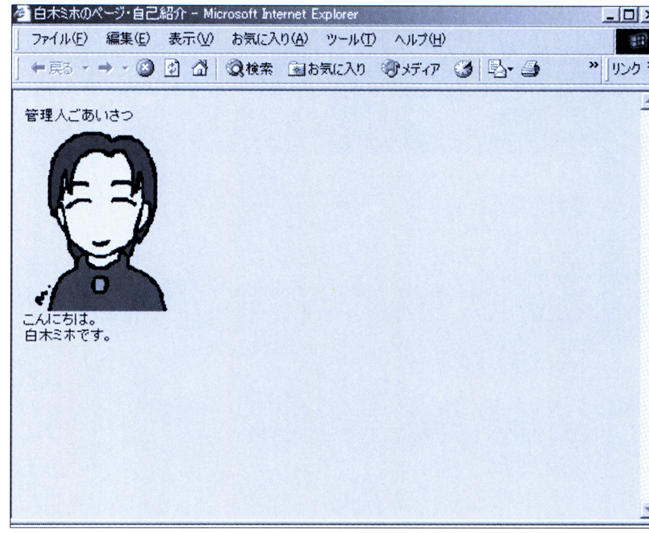
ではガゼンやる気が出てきたところで，いちばん簡単なホームページを実際に作ってみましょう。

第４図は文と画像の入ったいちばん簡単なホームページ，**リスト１**はその設計図のhtml文書ファイルで，「ソース」と呼ばれているものです。

ホームページは「タグ」と呼ばれる記号を使ったこのソースでできています。作成支援ソフトユーザーのなかには，このタグの知識がまったくないままにホームページを作っている人もいるようですが，いくつかの基本は覚えておかないと上級編に進めないので，腹をくくるべき



《第4図》いちばん簡単なホームページ。簡単すぎて何の芸もありませんが



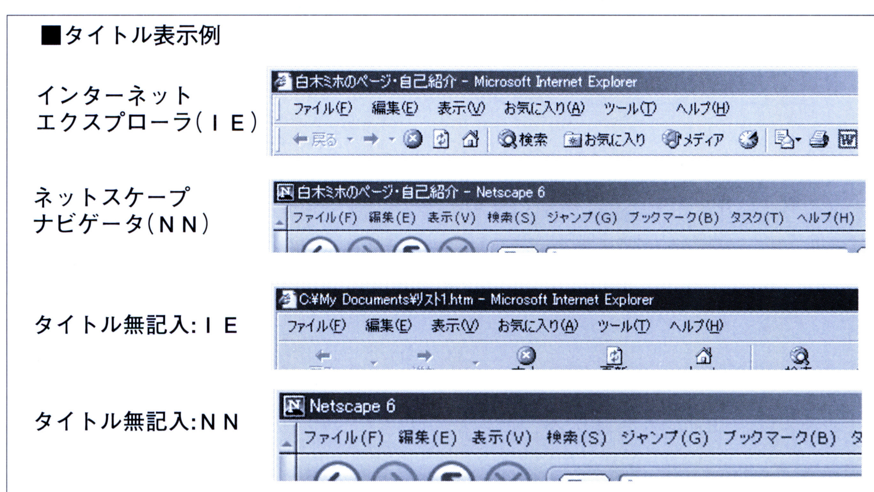
《第5図》技アリ！ユニバーサルデザインのホームページ。見た目は第4図と変わらない？けど，工夫がいっぱいです

```
<html>
<body>
管理人ごあいさつ<br>
<IMG src="shiraki.gif"><br>
こんにちは。<br>
白木ミホです。<br>
</body>
</html>
```

《リスト1》第4図の画面のリスト。この画面を表示するための最小限のリスト。ソースはホームページの設計図

```
<html>
<HEAD>
<TITLE>白木ミホのページ・自己紹介</TITLE>
</HEAD>
<body>
<body bgcolor="#ffcccc">
管理人ごあいさつ<br>
<IMG src="shiraki.gif" alt="100万ドルの笑顔"><br>
こんにちは。<br>
白木ミホです。<br>
</body>
</html>
```

《リスト2》第5図の画面のリスト。第4図と画面は似てるけどリスト1とは全く違うソース。技が詰まっています



《第6図》ブラウザのタイトル表示。見た目も分かりやすく，音声リーダ使用者には頼みの綱。このたった1行が使い勝手を変えます

でしょう。

去年連載した『iモード・ホームページ講座』を購読してくれたキミにはダブってしまいますが，まあ復習と思ってやってください。

まず，タグは基本的に「ペア」になっています。

リスト1を見ると<html>に対して</html>，という具合に「/」(スラッシュ)のなし・ありで「組になっている」のが分かります。
，<hr>など，単独で稼動するタグもいくつかあるのですが，基本的にペアが「ここからここまで」と命令実行の範囲を挟みこむことで仕事をします。

これがなければホームページじゃない，いちばん根本となる<html>と</html>のペアは，「このプログラムはhtmlテキストファイルで，ここから始まります」そして「ここで終わります」と宣言しています。<body>と</body>のペアもそれぞれ「ここから本文です」，「ここまでが本文です」と示しています。ホームページの内容はこの2種類のタグには挟まれた部分に命令を書くことによって認識，実行されるのです。

<img src="shiraki.gif">は画像ファイルをここに呼び出しなさいというタグです。「" "」に挟まれているのはその画像の名前で，ファイルの種類を示す「拡張子」(ここでは「.gif」)が付いています。

タグは半角英数字で書きます。大文字小文字は認識しませんが，画像ファイルやhtmlテキストファイルの名前は別個に

認識するので注意が要ります。さらに，ソース内で改行しても実際の画面には反映されないので，改行したい場所には
というタグを入れます。

これでいちばん簡単なホームページはできあがりです。

ユニバーサルデザインはここに一工夫アリ

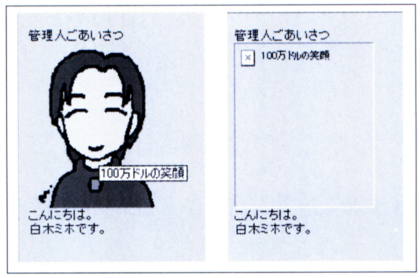
ではこのいちばん簡単なホームページに，その上を行くユニバーサルデザインを施しましょう。

第5図は一見，**第4図**と変わりませんが，ユニバーサルデザインに基づいた「技アリ」ホームページになっています。

①タイトル情報を入れる

画面左上の，その名も「タイトルバー」にちゃんとタイトルを入れましょう（**第6図**）。

見てわかることも重要ですが，音声リーダの使用者にとっては自分が今どこのページにいるかを知るための重要な手がかりだからです。聞いてわかりやすいタイトルの書き方としては，トップページに「白木ミホのページ」と入れたら，次は「白木ミホのページ・自己紹介」などのようにつけると，「どこのホームページの何か」がはっきりして階層も分かりやすくなり，ネットサーフィン向きになり



《第7図》音声リーダだけじゃない，ALT属性は画像の説明文。IEでは画像にカーソルをのせると説明文が表示される。リンクが切れていてもわかりやすい

ますね。

タグは，<body>（本文）の前に位置する<head></head>の範囲に書きこみます。<title> </title>で挟まれた部分がタイトル情報の欄になります。

「でもタイトル情報ってフレーム使うと出ないんでしょ？」と，すでにホームページの知識を持っている人は鋭い疑問を持つかもしれません。確かに，画面を分割して複数のページを表示する「フレーム」と呼ばれる画面構成の方法では，このタイトル情報は画面上に現れなくなります。ですが，音声リーダではフレームによって仕切られた画面の中に，何と何が表示されているかをこのタイトル情報によって示すので，画面に「見えない」場合でも情報は書きこんでおいてください。



**《第8図》文字間に空白を入れない。リスト上で「管□理□人□ごあいさつ
」となっていると，音声リーダは別々の単語と認識してしまうのです**

②画像にALT属性をつける

ALT属性とは，画像に説明文をつける機能です。画像表示を命令するタグの中に「alt="100万ドルの笑顔"」と書いてあります（**リスト2**）。この「"　"」の間に挟まれた文字列が音声リーダに読まれることによって，目の不自由な人でも画像を「聞く」ことができるようになるのです。

この機能は，現在では少なくなりましたが画像表示のできないブラウザのユーザーのために，画像の内容を説明するという重要な役目を持っていたので，旧来のユーザーには知られている機能です。また，画像が何らかの理由で表示されなかった場合にも，説明は表示されるので親切です。ブラウザによっては，画像領域にカーソルを乗せると文字がポップア

コラム　Web作成支援ソフトの功罪　白木ミホ

ホームページと言えば，ちょっと前まではタグをせっせと自力で打ちこんで画面表示で確認し，間違っていればその箇所をチェックしてという地道な作業でしたが，それを一変させたのが「ホームページ作成支援ソフト」の存在です。

ＩＢＭ社の『ホームページ・ビルダー』やマイクロソフト社の『フロントページ』など，高度な機能を持ちながらも扱いやすい支援ソフトの登場は，タグを全く知らない人にもホームページの作成を可能にしました。その結果，ホームページに作り手として参加する人の年齢層が格段に広がったのは，ソフトの大きな功績でしょう。

しかし，一方でソフトがもたらした新しい問題もいくつかあります。

一つは，「タグ」を理解し覚える機会がなくなったことでホームページの構造が理解できず，そのため作成支援ソフトが持っている問題点を超えられないということです。

現行のソフトでは，書き換えや削除を繰り返しているうちに消しきれずに残ってしまう「ゴミタグ」が誤表示の原因になることもあります。タグを理解できていれば不要なものを自力で削除できるのですが，知識がない場合はそれができません。また，掲示板やゲームなどのＣＧＩを設置する時にもタグの知識は必要ですが，知識がなければそれらの上級技も使えません。

もう一つの問題は，高度な機能が手軽に使えすぎるためにユニバーサルデザインから遠ざかっている，ということでしょう。高度すぎる機能がユーザーを限定してしまうことは本文でも述べましたが，日本語版登場も間近の「第３のブラウザ・『オペラ』」が参入すれば，さらに問題は複雑になるかもしれません。

有料ソフトは，どうしても制作会社の「色」を引きずります。頼んでもいないのにフォントが「ＭＳ明朝」（ウィンドウズの搭載書体）になってしまうと言って怒っても，そのソフトがマイクロソフト社製なら，ある意味どうにもなりません。「ビルダーはマックに冷たいからねえ」とぼやくユーザーの声はおなじみです。

これらの問題点を解決するには，やはりタグの機能を覚えることです。そして自力で特化を回避するしかするしかありません。がんばりましょー！

ップ表示されるものもあります（**第７図**）。

書き方のコツは「写真」のような抽象的なものでなく，「白木の写真」のような説明を心がけること。といっても「右手を上げて笑顔を見せている白木ミホの写真」では長すぎてうっとうしいので，そこは簡潔に。

さらに，「alt="こんにちは"」などのように書くと，音声リーダにかけた場合，「管理人ごあいさつ…こんにちは…こんにちは…白木ミホです」と前後の文字情報と重複して間ぬけなことになるので，それも注意しましょう。

③文字の間に空白を入れない

レイアウト＝見栄えをよくするためによくやってしまう手ですが，文字と文字の間に空白を入れるのは，これもまた音声リーダ泣かせです。空白が入ったことでリーダは一つの単語がそこで終わったと認識してしまうため，**第８図**のような出力をしてしまうのです。これでは何が何だか全然分かりません。見栄えは別のところで追求することにして，文字は切らないのがユニバーサルデザインの「できる」証です。

④背景と文字にはコントラストをつける

これは色弱の人に対する配慮に限らず，よりよいホームページデザインを心がけるすべての管理人が注意すべきことでしょう。黒の背景に深緑の文字を書いたり，はたまたパステルの背景にパステルの文字を乗せたりすれば，誰だって読みにくいに決まってます。あるいは黄色と黄緑のような近すぎる色相や，真っ赤な背景に真っ青の文字といったサイケ調も疲れ目気味のネットサーファーには大敵です。ここだけの話，真っ白な背景も徹夜明けの目にはツライものがあります。

そんな時は<body bgcolor="#ffcccc">のタグを使って目に優しい背景の色指定をしましょう。「"#　"」の中に0～9，A～Fの計16個の記号を使って16進法で色を指定できます。各色についての細かい説明はここでは省きますが，カラーチャートや実際の画面を参考にしてください。

このように，たった4点について見ただけでも，現状のホームページがいかにバリアフリーにほど遠いかがわかると思います。

しかし同時に，このたった4点を加えるだけでも，キミのホームページは「技アリ」になるのです。

技アリへのさらなる道

今回はユニバーサルデザインとは何かということと，ホームページ制作におけるその基本中の基本についてご紹介しました。

次回以降では文字や画像など具体的にテーマをしぼりつつ，ちょっと目立つ小技や，ユニバーサルデザインの方法をマスターしていきましょう。ではまた！

コラム２　第3のブラウザ「オペラ」日本語版登場！　白木ミホ

インターネット人口の約70パーセントが「インターネットエクスプローラ」，約25パーセントが「ネットスケープ」ユーザーと言われるブラウザの世界で，「第3のブラウザ」として注目されているのが，フィンランドの「オペラ・ソフトウェア社」が作り上げたブラウザ『オペラ』です。

その待望の日本語版がようやく登場し，さる２月８日から同社ホームページで無料版の配布が，3月下旬からはライセンス販売が開始され，日本でのユーザー拡大に乗り出しています。

オペラの特徴は何といっても3.25MBというコンパクトサイズです。エクスプローラがWin98で12MB，ネットスケープが同26MBですから，Java対応版でも10.8MBというのは驚きです。また，今回のウィンドウズ版発売以降，マック，リナックスなどの他ＯＳ版も発売される間口の広さと，古いマシン/小さいメモリでも動作する使い勝手の良さで，モバイル分野での活躍も期待されています。

というわけで，さっそく無料版をダウンロードしてみました。

「速い」が売りだけに，確かにテキストの表示はサクッと完了しますが，背景画像の立ち上がりは若干遅い印象がありました。優先順位ということでしょうか。

表示方式は1ブラウザ内に１サイトのシンプルウィンドウと，複数サイト表示のマルチウィンドウの２種類ですが，シンプルウィンドウでも「新規ウィンドウ」を一つのブラウザ内のバックグラウンドにクリック一つで立ち上げられ，並んだタイトルをクリックするだけで切り替え可能なので，エクスプローラユーザーには新鮮な快適さがあります。

ユニークな使い方としては，機能を盛りだくさんにしたマウス動作のおかげで，ネットサーフィンのかなりの手続きがマウス一つでできるということも挙げられます。Ctrl＋マウスホイールのコロコロで画面のズームイン/アウトができるあたり，徹夜明けユーザーにもやさしいお茶目さを備えています。

現状で気になる点は，無料版に漏れなくついてくる広告表示です。広告あっての無料化とはいえ，一方でポップアップ広告を消去する「フィルター」機能を常備しつつ，一方では，「閉じる」ボタンのない広告を，入金するまで見せ続けるというしくみは，ユーザーにしてみれば矛盾とも見えます。日本版はスタート記念価格3,800円だそうですが，無料版には「広告を見る」という代価が要求されているのです。

その穴埋めとばかりに，広告は少しでもユーザーのニーズに合うよう，年齢，性別，趣味などのカテゴリでの取捨選択が可能になっています。もちろんそれらの情報を故意に収集するものではないという但し書き付ではありますが，良くも悪くも「広告」が目立つあたり，ネット世界の最先端ということでしょうか。

それから，画面表示サイズを800×600で設定したモニタで，初期設定のオペラを立ち上げると，いきなり画面の３分の１近くがツールバーと広告で埋まります。いらないツールをばっさり非表示にし，「設定」で「ツールバーを折り返す」のチェックを外せば，他ブラウザなみのスペースに納められますが，ちょっとびっくりします。

ともあれ，シンプル・すっきりがかわいいオペラの，日本での活躍を見守りたいと思います。

オペラ日本語サイトのアドレス
http://www.jp.opera.com/index.html

パソコンの心臓部　マイクロプロセッサ誕生30周年企画

CPUの歴史と未来をさぐる!
プロセッサGHz時代到来!

文・絵／くりひろし

夢の2GHzマシン,10万円以下は当たり前〜!?

Dr.D：ほうっ。

つぐ美：どうしたんですか，Dr.D？タメイキなんかついちゃって…。

Dr.D：いやな，マイクロプロセッサ誕生から30年と同時に，プロセッサの動作周波数（クロック）も2GHzの大台に乗ったんじゃなぁ，と。

つぐ美：ああ，そういえば…年末にスベりこみで2GHzの大台に乗ったんでしたっけ？ 2GHzのプロセッサ搭載のパソコンってさ，今ならAthlon XP 2000+のパソコンでもPentium4 2GHzでも，ディスプレイ抜きで10万円以下で作れますねぇ。デフレはイヤだけど，パソコンが安くなるぶんには歓迎だわぁ。

Dr.D：かぁーっ，嘆かわしい，安くするにはメモリがDDRじゃないSD-RAMだとかビデオがオンボードじゃろ？それじゃあ，プロセッサの性能を発揮しきれんじゃろうて…フィアットにターボエンジンを載せるようなもんじゃ…って，わからんか？

つぐ美：…それってケッコウ速いじゃない（笑）…とにかく，それだって，1GHzマシンよりはずっと速いでしょう？安い，速いがイチバンよぉ！

Dr.D：うーむ，日本経済をなんだと思っておるのかのう。パソコンメーカーだって赤字やらなにやらで大変じゃというに…。

つぐ美：ふふん，お年寄りは過去の栄光にすがりたがるのよねん。かつてのパソコンメーカーの二強もすっかり崩れて，いまやＡＶ屋さんのパソコンがメイントレンドになっちゃったんじゃなくて？

Dr.D：か，過激なコトを…販売台数ベースなら，まだかつての二強のワン・ツーなんじゃけどな。それにしても，ここ１〜２年のパソコン業界って激動すぎて，もはやワシにはついていくのがやっとじゃよ。

つぐ美：んー，そういえば，もうあれから３か月もたつから，そろそろ新型の，もっとクロックの上がったプロセッサが発表になってもいいですね。

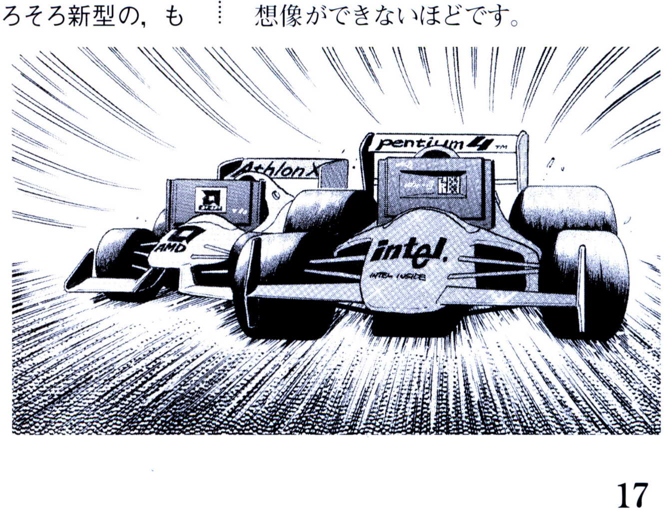
Dr.D：…カンタンに言ってくれるなぁ。

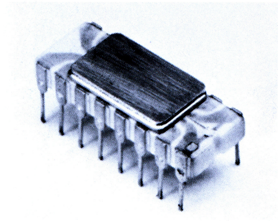
つぐ美：…カンタンじゃなかったんですか？

2001年末，とうとうパソコンのプロセッサの動作クロックは2GHz（20億ヘルツ），すなわち１秒間に20億回の動作（かならずしも命令とは限りません が）を実行できるようになりました。動作クロックは1971年の初代のi4004で108KHz（10万8千回の動作）でしたから，2GHzといえばその1万8千5百倍強です。使われているトランジスタの数，集積度だって2万倍…，単純な演算性能比で見れば実に4万5千倍以上の性能になったのです。

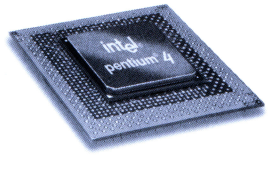
人類史上，これだけの性能向上があったものは，マイクロプロセッサ以外には，宇宙航空機関連と大量殺戮兵器くらいなものです。クルマなんかでは，Ｔ型フォードから100年たっても最高速度はせいぜい20倍くらい（時速20キロ→時速400キロ）なのですから，マイクロプロセッサの性能向上度合いが，いかにすごいものであるかは，にわかには想像ができないほどです。







▲1971年に発表された「i4004」。動作周波数は108KHz。



▲2001年に発表された「Pentium4」。動作周波数は2.0GHZ。

初のマイクロプロセッサVS. 最新2GHzクラスプロセッサ		
プロセッサ	i4004	Pentium4
発表時期	1971年	2001年
動作周波数	108KHz	2.0GHz
集積度	2,300	42,000,000
BM性能指標*	1	45,500

*BM性能指標とは，各プロセッサメーカーが発表している各種の性能指標や性能比較データを元に，筆者が独自の計算によってi4004を"1"とした時の性能を示しているものです。

人類の夢，1GHzを突破した日

Dr.D： しかしな，つぐ美くんよ。

つぐ美： はい？

Dr.D： 本当に感動したのは，最初に1GHzを突破したときじゃな。2GHz突破も確かに記念すべき出来事だったかもしれなかったが，ありゃ人類が初めて宇宙に行ったときに匹敵する感動じゃよ。忘れられんなぁ，映画「ライトスタッフ」のあのシーン！アメリカのな，初の有人宇宙飛行をどうするって会議の場にだな，伝令が入ってくるんじゃよ，息せき切らして！そんで，「ガガーリンです！」って！

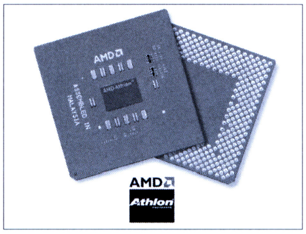
つぐ美： あ～，これだよ。始まったよ，昔話…。

Dr.D： 1960年代，冷戦中の米国とソ連（ロシアは当時，ソビエト連邦の一国家だった）が，国家の威信を賭けてつばぜり合った，あの宇宙開発競争の黎明期にも似た独特の雰囲気を，AMDとintelのプロセッサ1GHz競争に見た思いなんじゃよ～AMDが1GHz Athlonを発表した時のニュースリリースで"AMD Athlon Processor Rockets to 1 GHz"ってタイトルをつけるワケじゃよ。

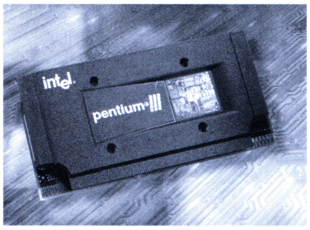
AMDとintelの，1GHzプロセッサは，どちらが先に発表できるか，お互いのメンツを賭けた戦いでした（ここで間違いないようにはっきりしておきたいのは，1GHzを突破というのは「市販品として量産できる」状態での1GHz突破，です。研究機関での，たった1個しかない特注品のプロセッサでは，たとえ何GHzになろうが，パソコン業界から見れば何の得もありません）。

技術的にはすでに，1999年2月にintelはPentiumⅢで1GHzを突破できる技術実験を公開していました。一方のAMDはintelに遅れる事1年，2000年の2月にようやくAthlonで1.1GHzHzが達成できた事を公表しています。おなじ頃，intelはすでに次のステージとして1.5GHz達成を公開していましたから，業界やジャーナリストの間では，当初はAMDは2000年5月に発表，Intelは2000年8月に発表の予定，と見ていました。みんなIntelが先に1GHzを発表するだろう，と思っていたのです。

本当はこの時期，どちらのメーカーも，世界初の（＝人類初の）1GHzプロセッサの栄光を勝ち取ろうと，業界筋への情報リークなど，情報操作合戦になっていました。実際のところ，すでにどちらも量産は可能な状態になっており，発表日を決めるだけになっていたのです。しかし，うっかり発表日を公にすると相手に先を越されてしまいます。まさにスパイ大作戦，エロイカより愛をこめて，の世界が繰り広げられていた…のは想像の域を出ませんね。結局，Intelが発表日を決めた事をキャッチしたAMDが，わずか2日前の日に，1GHz版Athlonを公式発表し，世界初の2GHz動作可能なプロセッサを世に送り出したのでした。史実はこうです。

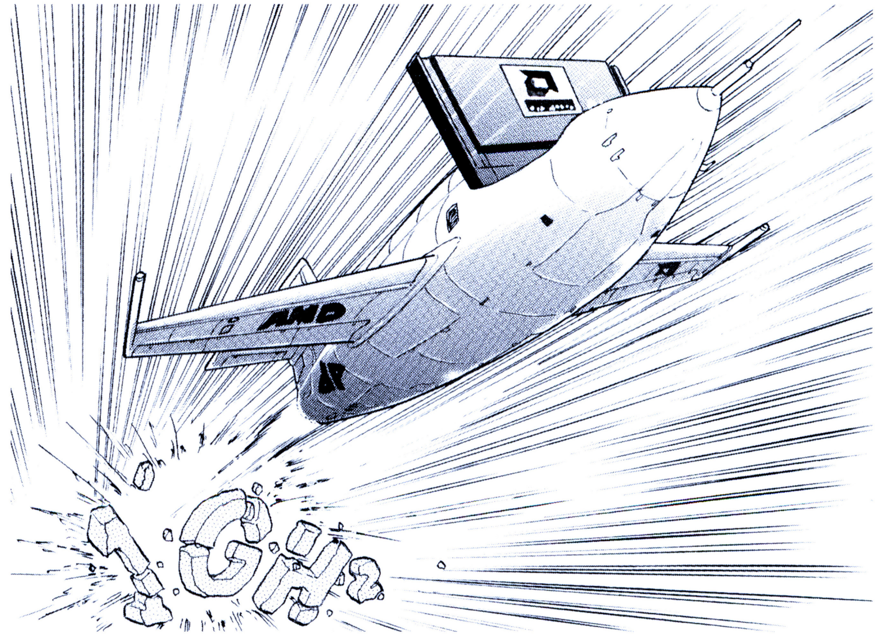


▶AMDの「Athlon」はintelに先がけ1GHzを実現。

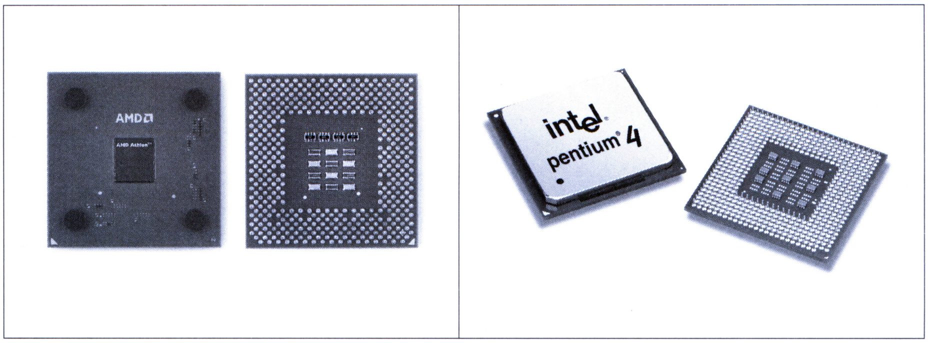


▶1GHzを突破した「PentiumⅢ」。

- 2000年3月3日　Intel社が1GHz動作Pentiumを発表すると情報が流れる
- 2000年3月5日　AMD社が1GHzプロセッサを発表すると情報が流れる
- 2000年3月6日　AMD社，1GHz動作のAthlonを発表
- 2000年3月8日　Intel社，1GHz動作のPentiumⅢを発表



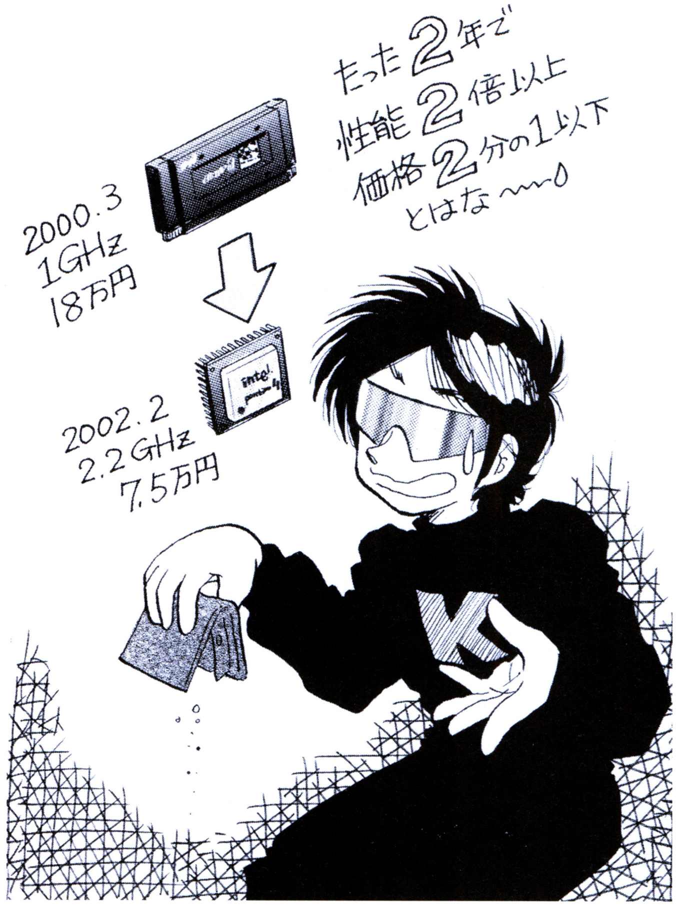
世界初の称号を手に入れたAMDはよほど嬉しかったのでしょう，当時のAMDのCEOは音速の壁になぞらえて，"Just as the achievement of Chuck Yeager signaled the beginning of a new era in aviation, the 1GHz processor ushers in a new era of information tech nology.（ちょうどチャック・イェーガーの偉業が航空機の新しい時代の始まりを知らせたように，１GHzプロセッサはＩＴ時代の新しい時代の到来を告げた）"と，1GHzのAthlonを発表した際にコメントしています。



▲現在主流のAMD「Athlon XP」(左) とintel「Pentium4」(右)。すでに2.0GHz時代に突入している

Intel vs AMD プロセッサ対決：１GHz突破までの攻防

1999. 2	Pentium Ⅲ 技術デモ：1GHz
1999. 8	Athlon デビュー 650MHz,600,550,500MHz
1999.10	Athlon 700MHz
1999.10	PentiumⅢ 733MHz
1999.10	Athlon 技術サンプル：900MHz
1999.11	Athlon 750MHz
1999.12	PentiumⅢ 750MHz
2000. 1	PentiumⅢ 800MHz
2000. 2	Willamette(後のPentium4) 技術デモ:1.5GHz
2000. 2	Athlon 技術デモ：1.1GHz
2000. 2	Athlon 800MHz, 850MHz
2000. 3	Athron 1GHz, 950MHz, 900MHz　AMDが一番乗り
2000. 3	PentiumⅢ 1GHz, 850MHz　Intelはわずか２日後



つぐ美： チャック・イェーガーねぇ…？ピンとこないわね。

Dr.D： だーかーらー,「ライトスタッフ」くらい見ておかんかね？ちょっと脱線じゃが，音速の壁突破ってのはこういう話じゃ。「かつて航空機にとって,『音速の壁』が存在していました。秒速340m，時速にして1,224ｋｍ，F1マシンの４倍の速度…マッハ1.0に近づくほど，航空機は激しい揺れと，何かにぶつかるような圧力,『大空に棲む悪魔』によって，コントロールを失い，空中分解したのでした。1947年10月，第二次世界大戦のエース，チャック・イェーガー大尉は最新鋭試作機ベルＸ－１によってついに音速の壁を突破することに成功したのです」。

つぐ美： ふーん，なんか命がけでカッコいい話だったのね。

Dr.D： ワシに言わせれば，某プロジェクトＸで取り上げて欲しいくらいじゃぞ。地上の星座，ええのう。テーマ思い出しただけで泣けるぅ。

つぐ美： うーん，さすが高度成長時代に働き盛りだったオヤジ族だなぁ〜。

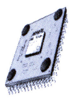
影： わ，私だって好きだよ！技術者マインドをもった人間にはわかるんだよぉ！

Dr.D： わかってくれるか！？め，めずらしいのう,影君と意見があうなんて…。

つぐ美： ホントにそうかしらねぇ…なんか，若さがないっつーか，過去の栄光を追い求めてるっつーか…。

影： いやだから私は若いって！

さて，実際に市場に出回ったのも1GHz Athlonの方が早く，発表から１か月もしないうちに販売されました。初登場時の市場価格は平均１８万円程度。一方のPentiumⅢ 1GHzは７月になってようやく市場に出回り，平均１３万円程度。…ちなみに，２年もたたずに今やPentiumⅢもAthlonも，後継のPentium4とAthlon XPに取って代わられ，最低動作クロックも1.3GHzで価格も１万5〜6千円…今年の夏には１GHz以下の新製品は絶滅しそうないきおいです（なんて書いてるうちに，一部の1GHzプロセッサが製造終了ですと)。ホントにすごいですね。



高速化の道はイバラの道

つぐ美： しかし，なんでそんなに動作クロックにこだわりますかね。

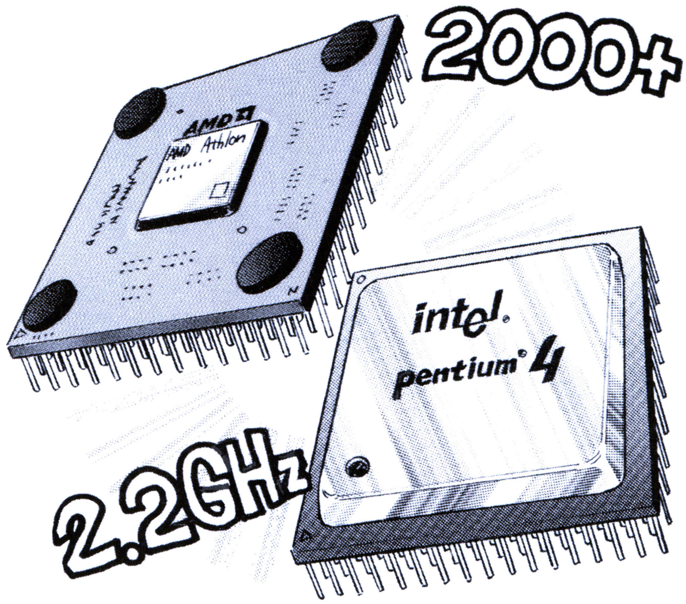
影： そりゃ，同じプログラムを走らせるんなら，早いほうがいいに決まってる

じゃん？クロックが高い＝速い，ってことさ！最近は高速プロセッサの値段も安いからあまりはやらないけど，ちょっと前まではパソコンの改造って言えばクロックアップが定番だったよね。初期のCeleronなんてさ，クロックアップ耐性が高くって，２割３割アップは当たり前〜だったよ。

つぐ美：そんなんばっかやって，やりすぎてプロセッサを壊しちゃうから，「パソ破の影」なんていわれちゃうのよ！

影：ううう，ミニ四駆だってベイブレードだって，改造こそが勝利への道なんだよぉぉぉ〜。

Dr.D：気持ちはわからでもないが，やれやれ…。



プロセッサの性能を比べるには，本来は初期のZ80や6809の時代のように，プログラムが書き易いとか，周辺チップがそろっているか，などいろいろなファクターがあります。しかし，近年のパソコンではMacintosh系をのぞいては，ほとんどすべてが80386からはじまる，いわゆるIntel x86系互換のプロセッサです。中身が厳密に同じわけではありませんが，同じコードが走る以上は，スピードと価格がプロセッサの競争の主力になるわけです（本当はマルチメディア命令の有無とかもあるんですが，結局お互いが同じ機能を搭載しあうので，ほとんど比較にならない）。プロセッサのスピードアップ，処理速度のアップに直接的に効果があるのは，やはり動作クロックの向上です。同じプロセッサなら，100MHzよりも200MHzの方が単位時間内に多くの処理ができる＝速い，にきまっています。しかも，パソコンユーザーにとっても性能の指標としてわかりやすいのです。そこで，K5,K6 vs Pentium, PentiumⅡにはじまるプロセッサ戦争は，そのままクロックアップ競争になっていました。

１GHzプロセッサへの到達のためには，いくつもの技術的なハードルを乗り越える必要がありました。高速化すること，すなわち動作クロックを上げる事による最も大きな問題は，発熱量でした。結局，作りの同じプロセッサに対して，単純に動作クロックを上げても，発熱量が大きくなって，そのうちに壊れたり，誤動作をしたりで実用にはなりませんでした。そこで，初期のクロックアップ改造がブームだった頃は，プロセッサの出す熱を発散させるために，より大型のヒートシンクに取り替えたり，ヒートシンクだけのものに冷却ファンをとりつけたり…あげくの果てには水冷式の冷却装置まで売り出される始末でした。

本質的にプロセッサの性能向上を図るためには，プロセッサの構造自体を変えていく必要がありました。プロセッサが発熱するのは単純な物理現象であり，プロセッサの回路の部分を通り抜ける電気の抵抗のためなのは，読者のみなさんにも理解できるでしょう。わずか数ミリ四方のシリコンでできた集積チップの中に，数千万個のトランジスタが生成されているようなマイクロプロセッサでは，プロセッサを構成するトランジスタとその回路に流れる電流の量を変える事が，発熱量を下げる事につなげられます。これだけマイクロな電子回路では，回路の線，間隔が細いほど発熱量を減らす＝消費電力を減らす事ができるのです。つまり，プロセッサをどれだけ細かく製造できるか——これを製造プロセスといいます——が，高速化の決め手だったわけです。初代のPentiumⅢやAthlonでは，プロセッサの中のトランジスタの回路の幅＝製造プ

Pentium vs Athlon 開発コード名と製造プロセス

プロセッサ	開発コード名	製造プロセス	特徴
初代PentiumⅢ	Katmai	0.25μm	950万トランジスタ
二代目PentiumⅢ	Coppermine	0.18μm	1GHz版／2,810万トランジスタ
三代目PentiumⅢ	Tualatin	0.13μm	現在の製造技術のベース
初代Pentium4	Willamette	0.18μm	4,200万トランジスタ
二代目Pentium4	Northwood	0.13μm	2GHz版，現在の主力／5,500万トランジスタ
初代Athlon	K7	0.25μm	2,200万トランジスタ
二代目Athlon	K75	0.18μm	改良版
1GHz版Athlon	Magnolia	0.18μm	本来700〜800MHz用，1GHzは無理矢理
四代目Athlon	Thunderbird	0.18μm	1GHz正式版／3,700万トランジスタ
Athlon XP	Palomino	0.18μm	2000+，現在の主力／3,750万トランジスタ

ロセスが0.25 μ m（4千分の1ミリ）でしたが，1GHzを達成するためには，ここから約2/3，0.18 μ mまで小さくする必要があったのでした。ちなみに，2.2GHzを達成した最新型のPentium4では実に0.13 μ mです。トランジスタの数も4倍以上ですから，高速化と高集積化の両方が同時に達成され，より速く，より高性能に進化し続けているのが，今日のマイクロプロセッサの製造技術なのです。

ちなみに，消費電力の低下と，高集積化すなわちプロセッサの本体サイズ＝ダイサイズの小型化は，単なる高速プロセッサだけではなく，小型軽量，低TDP(熱設計電力：Thermal Design Power)をもって「是」とする，モバイル機器用プロセッサへも大いに恩恵をもたらす事になりました。おかげで今や，ちょっと高級なノートＰＣならば動作クロックは1.2～1.5GHz…ほんのちょっと前まで，ノートPCといえばCeleron 400MHzぐらいが定番，という感じでしたから，つくづく進歩のスピードを感じさせられます。

そして二大プロセッサメーカの競争は続く

つぐ美：1GHz突破したら，さぞかし性能アップ！したんでしょうねぇ。ユメの1GHzだったワケだし。
Dr.D：んにゃ，実のところ，1GHz突破したからといって劇的に性能が向上したワケじゃなかったんじゃな。当時のベンチマークではAthlon 1GHz 対 850MHzの場合，おおむね10：9の性能比じゃった。すなわち，動作周波数の比率ほどの性能向上は無かったって事だ。
影：そら，そのハズですよ，1GHzプロセッサって，専用の全く完全な新製品じゃなくって，850MHzとか900MHzとかの製品の質を上げて，1GHzでも動作できるものを1GHz版としているからですよね。よくある方法なんだけど…。
Dr.D：うむ，わかっとるのう。「歩留まり」という言葉が存在するのはそのためじゃな。10000個プロセッサを作ったうちの8000個が1GHzで動作できるなら，歩留まりが8割って事になる。
つぐ美：…な，なんかダマされてるようなな…
影：そーゆーもんなんですよー。
Dr.D：あのな，誤解のないように言っておきたいのだが，製造ライン自体はあくまで高速な方を目的に作っておるからな。低い性能目的から速いのを選別するのではなく，高い性能目的のほうから，性能不足のものを落としとるのだよ。

Intel vs AMD プロセッサ対決：１ＧＨｚ突破後の優位戦争

2000. 8	Athlon1.1GHz
2000. 8	PentiumⅢ 1.13GHz
2000. 8	Pentium4 技術デモ：2GHz
2000.10	Athlon 1.2GHz
2000.10	Pentium4 1.4GHz
2000.11	Pentium4 1.5GHz
2000年末	早くも価格破壊競争
2001初頭	FSBの高速化
2001. 5	Athlon 1.4GHz
2001. 5	Mobile Athlon 4 1GHz　ついにモバイルも1GHzを突破
2001. 6	Pentium4 1.8GHz
2001. 7	Mobile PentiumⅢ-M 1.13GHz
2001. 8	Duron 1.0GHz　バリュー向け1GHz突破もAMDの勝ち
2001. 8	Pentium4 2MHz, 1.9GHz　2GHz突破はIntelの勝ち
2001. 9	Celeron 1GHz,1.1GHz
2001.10	Duron 1.1GHz
2001.10	Athlon XP 1800+ (1.53GHz)
2001.10	PowerPC G4 1GHz　ついにiMacも1GHzマシン時代に
2001.11	Athlon XP 1900+ (1.6GHz)
2001.11	Duron 1.2GHz
2001.11	Celeron 1.2GHz
2001.12	Pentium4 2.2GHz　現在の最高動作周波数
2001.12	Athlon XP 2000+ (1.67GHz)　性能的には2GHz級に
2001.12	Mobile Duron 1.1GHz, 1GHz
2002. 1	Celeron 1.3GHz
2002. 1	Mobile Athlon 4 1500+
2002. 2	Duron 1.3GHz

さて，気になる1GHzプロセッサ同士の性能対決はどうだったのでしょうか？当時のベンチマーク対決では，おおむね平均すると2～5％ほどPentium 1GHzの方が高速でした。しかし，初代1GHz Athlonはそもそも1GHz目的のプロセッサではありませんでした。PentiumⅢの1GHz突破一番乗りへの対抗上，旧式コア（プロセッサの核部分）の製品から無理に1GHz版をひねり出していたからです。本来の1GHz対応プロセッサは，数ヶ月後に発表される，開発コードThunderbird（なんて響きの良いコードネームでしょう！）と呼ばれた新しいバージョン(L2キャッシュを内部に統合，増設した高速版）だったのです。それ以降のAthlonは，同じ動作クロック数であれば，常にPentiumを凌ぐ性能を発揮するようになります。これが，後の「＋(プラス)」と呼ばれるAthlon XPシリーズへの伏線になっていたとは当時だれも思わなかったのですが…。

競争はテクノロジーの進歩の原動力です。本当のところ，AMDがいなければ，K6が，Athlonがなければ，まだ世

の中に1GHzプロセッサは出回っていなかったかもしれません。第二次世界大戦と米ソの冷戦，軍拡／宇宙進出競争がテクノロジーの進歩を大幅に加速したように，ライバルなかりせばプロセッサの超進化もなかったことでしょう。宇宙競争はソ連の崩壊と同時に歩みが停滞しています。わたしたちパソコンユーザーは，これからも二大メーカーの競争が続くことを願いましょう。

安い速い！価格対性能比は抜群な２ＧＨｚプロセッサ

つぐ美：あんまり適切ではないけれど，最新のPentium4 2.2GHzって，だいたい75,000円くらいでしょ？つーことは１円あたり29KHzなワケよ。

影：あ，言いたい事わかったぞ。１年前のPentium3 1GHzだったら，約130,000円だから，7.7KHzくらいしかない，つまりコストパフォーマンスが１年で４倍だっていうこと？

つぐ美：あら，わかる？

影：なんか，主婦みたいな気のつき方だなぁ…。

つぐ美：つぐ美ちゃん18歳なのよ，それをいうなら奥さまは18歳，だわ！メダル取れなくって残念！

Dr.D：…そ，それは違う岡崎だろう…。あー，ネタのわからん読者諸君は，学校の先生に聞いてみてくれたまえ…。

Intel vs AMD プロセッサの市場価格状況（2002年2月現在）

ハイエンド向けプロセッサ						
最高周波数	Pentium4 2.2GHz	75,000円	←→	Athlon XP 2000+	38,000円
最低周波数	Pentium4 1.3GHz	20,000円	←→	Athlon XP 1500+	16,000円
バリュー向けプロセッサ					
最高周波数	Celeron 1.3GHz	15,000円	←→	Duron 1.3GHz	13,000円
最低周波数	Celeron 950MHz	9,000円	←→	Duron 1GHz	8,000円

というわけで，今やプロセッサは非常に安価でありながら，とびぬけて高性能なパーツです。しかし，その性能をちゃんと発揮させようと思うと，とたんに割高なメモリ，割高なマザーボード，高価なビデオボードが必要になります。結局，最高性能を出すためには，それなりにお金がかかるのはあまり変わりません。とはいえ，エントリーモデル，いわゆるバリューＰＣ向けの構成であれば，恐ろしく安く～ほんの4，5万円で基本性能としては十二分なパソコンを組み上げる事ができます。

かつては自作パソコンの高性能化には，まず安いプロセッサで一式組んで，あとからＣＰＵをアップグレードする手法が盛んでした。最近はマザーボードもプロセッサと同じくらいのスピードで進化しているので，古いPentiumⅢ 1GHzのマシンをPentium4 2GHzにするには，マザーボード＋メモリごと交換する必要があります。それでも十分安くアップグレードできるのですが，それよりも最初から高性能のプロセッサにしておいて，ビデオカードなどの周辺をだんだんいいものに変えていく方が良いかもしれません。

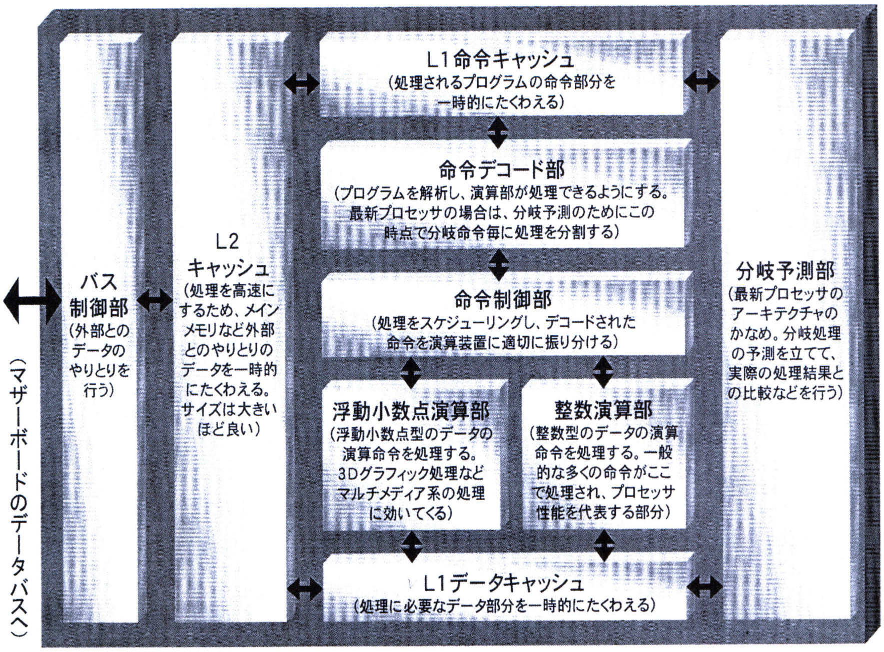


動作クロックだけが性能をあらわすものではない！

つぐ美：そういえば，最近のAMDのプロセッサって，直接ＧＨｚとかＭＨｚを名乗らないですね。1500＋とか2000＋とか…。

Dr.D：うむ，確かにもともとは同じ動作クロックなら同じ性能…つまり動作クロックの大きい方が性能が高い，ってことじゃった。が，実際のところ，低い動作クロックでも，メーカーが違えば性能が名前よりも高いってのはあるわけじゃ。まるで同じプロセッサではないんじゃからな。

つぐ美：それで，AMDにいわせれば，動作クロックの比較をされたくないので，Intelのプロセッサの動作クロックに匹敵するという意味で，ナントカ＋って名前にしてるのね。Athlon XP 2000+ で，Pentium4 2GHzに匹敵する性能を謳っているんだけど（実際にもほぼ同じ性能），実際の動作クロックは1.67GHzだって。



▲**最新型プロセッサの構造概要図**（Athlon,Pentium共に少しづつ違いがあります）

ADMは昨年10月に発表のAthlon XPから，プロセッサの動作クロック表示をやめ，対Pentium系の動作クロックを意識した性能指標を記すようになりました。また，対するIntelもしばらく以前から，これまで使っていたiCOMPのような単純なプロセッサ性能の比較をやめました。動作クロックは同じ系列のプロセッサでは非常に意味のある値ですが，いまや目的によって性能比較のためのパソコンシステムの構成も変えていく時代です。したがってIntelなどの性能比較評価も，オフィス業務向けとか，ゲーム向け，グラフィック処理向けなど，すべて別々のシステム構成で性能評価を行っています。

動作クロックの値だけで性能を比較できないのはAMDだけではありません。ベーマガ読者のみなさんだとあまり気にしていないかもしれませんが，Macintoshで採用しているPowerPCプロセッサでは，昨年の秋にようやく１ＧＨｚに到達しました。しかし，AthlonやPentiumのような根本的には古いx86アーキテクチャとはずいぶん違った進化を遂げているPowerPCでは，そもそも同じ動作クロックで比較される事に無理があります。

PowerPC G4 1GHzは2001年10月にようやく発表になり，この年明けのPowerMac G4で採用になりました。PowerPC G4ではVelocity Engineなる，128ビット演算機能をもつことで，32ビットの浮動小数点演算を同時に四つ，すなわち４倍のスピードでデータ処理できるのです。科学計算用の，いわゆるスーパーコンピュータで使われている技術を採用したPowerPC G4は，動作クロックで換算すると，さすがに４倍の性能とはいきませんが，命令や処理の種類によって２倍から３倍と，１ＧＨｚでも実にｘ８６系の２ＧＨｚプロセッサに匹敵する（画像処理など特定の処理によってははるかにしのぐ）性能を発揮できるのです。

次世代プロセッサでも熱い戦いが！

つぐ美：そういえば，そろそろ次の世代のプロセッサが出始めましたよね！64ビットプロセッサ！

Dr.D：うむ，まずはIntelの出したItaniumじゃな。まだ64ビットアーキテクチャのプロセッサは出始めたばかりなので，動作クロックもそれほど高くないが，現在のPentium4などの2GHzクラスを飛び越えるのは時間の問題じゃろう。

つぐ美：新しいプロセッサって，単に今までのプロセッサを速くしただけじゃないのよね？

Dr.D：もちろん，より高性能にするための仕組みがないわけではないぞ。

IntelもAMDも，次世代プロセッサはみな６４ビットです。６４ビットプロセッサでは，性能アップのために，動作クロックの向上よりもＩＰＣ（Instructions Per Cycle)の向上を図っています。Pentium4やAthlon XPなどではすでに分岐予測による高性能化を行っていますが，次世代では特にこの分岐予測性能を重視しています。ＩＰＣは要するに，１サイクルの動作でいくつの命令が実行できるか，です。命令を実行するパイプライン（たくさんの命令を続けて実行するために，パイプに流し込むような感じで処理するしくみ）の性能が高い事が，ＩＰＣの性能，すなわち分岐予測性能が高い事を示しています。また，メモリ，特にプロセッサに内蔵しているＬ２キャッシュなどの容量やアクセス速度も，プロセッサの性能アップに関わっています。

次世代プロセッサのアークテクチャ

メーカ	開発コード	アーキテクチャ
Intel	Mckinley	64bit, IA-64
AMD	Hammer	64bit, x86-64

では，今年は果たしてどこまでプロセッサ性能はアップするのでしょうか…。今後，年末までの筆者の予測では，せいぜい，Pentium4 2.5GHz/Athlon2500+/Mobile Pentium4-M 2GHz/Mobile Celeron 1.5GHzといったところでしょうか。今年はこれまでのAthlon XPやPentium4の高クロック化より，むしろ次世代プロセッサであるMckinleyやHammerが改良を加えられながら伸びていくと思われます。すなわち2002年は次世代元年！って感じでしょうか。

つぐ美：Hammerなんて，カッコいいわよね。ジャック・ハンマーか，範間刃牙かって感じですよね！いかにも強そうというか，カッコイイというか…。

影：やっぱ，俺様的には「俺がハマーだ！」ってカンジかな。

Dr.D：ううむ，なんかみんな漫画とかテレビの見過ぎだよな～～。

つぐ美：ん～～，誰もヒトの事は言えないようなぁ～～。

ベーシックプログラミング教室

BM基礎講座

第1回 プログラミングの第一歩

講師紹介

川上朝幸（Kawakami Tomoyuki）

1971年8月31日　大阪府生まれ



音楽系専門学校卒業後1993年（株）カプコン入社。サウンドエンジニアとして，大ヒット作"ストリートファイター2"，"ヴァンパイア"，"ロックマン"シリーズをはじめ"バイオハザード"，"ディノクライシス"など，多くのサウンド制作を経て2000年退職，翌2001年"スタジオ スターノーツ"を起業。サウンドエンジニアとしての活動を軸にプログラミング，パソコンスクール講師，ホームページ制作，ビデオ制作，アマチュア・インディーズバンドの支援活動など幅広い活動を行っている。

STARNOTES

http://www.starnotes.jp

basic@starnotes.jp

この講座では，Windows上で実行することができ，誰でも容易に入手することができるフリーソフトを利用しますが，ベーシックとして基本的なN88BASICに基づいた内容ですので，あらゆるベーシックで応用することができます。

☆フリーで登録されているベーシックプログラミング・ソフトをいくつか紹介します。

N88互換BASIC for Windows95

<http://www.vector.co.jp/soft/win95/prog/se055956.html>

ActiveBasic

<http://www.vector.co.jp/soft/win95/prog/se117283.html>

講座内ではこれらを互換BASICと呼ぶことにします。

みなさん，はじめまして！今月号から6回にわたってベーシックプログラミング教室の講師を担当させていただきます。どうぞよろしくお願いします。

僕がベーシックと聞いて真っ先に思い出してしまうのが，ファミリーベーシック（1984年に任天堂ファミリーコンピュータ（ファミコン）の周辺機器として発売されたもの）です。当時13歳だった僕は，周りの友人たちが，NECのPC6601，PC8801といった，いわゆるパソコンでプログラミングを組んだりゲームを楽しんでいるのに憧れよく遊びに通っていました。当時はまだまだ高価だったパソコンを手に入れる事ができなかった僕は，安価で購入できたファミリーベーシックを手に入れ，みなに負けじとプログラミングに励んでいたことを覚えています。

『マイコン・ベーシックマガジン』にもファミリーベーシックの記事や投稿プログラムが掲載されていたので，毎月購読していました。このころに覚えたベーシックですが今でもやはり頭の中にしっかり記憶されていて，サウンドエンジニアとしてカプコンに入社した後もとても役に立ちました。今も毎日の仕事の中でパソコンと接していますが，「ベーシックをやっっててよかったなぁ」と感じることがよくあります。みなさんもこれからプログラミングを覚えていくことで，きっとそう感じるときが来ると信じています。

BM基礎講座ということで，なるべくシンプルにわかりやすく進めていきます。サンプルプログラムや練習問題なども用意していますので，実際にプログラミングを行いながら進めてください。また6回の講座を通じて，ゲーム制作の雰囲気を誌面上で再現しながら簡単なゲームを制作していきます。みなさんもゲーム制作スタッフの一員としてがんばってください。

プログラミングというと，最初は難しそうでとっつきにくいイメージがあるかもしれませんが，この講座が終了するころには，自分で簡単なプログラムを組んだり，ベーシックマガジンに掲載されている先輩プログラマーの投稿プログラムが何をやっているのかを，ある程度理解できるようになるでしょう。明日のプログラマーを目指して，みなさんがんばっていきましょう！

プログラムを入力しよう！

コンピュータにさせたい仕事の手順を，コンピュータが理解できる命令で表現したものをプログラムといい，その命令をプログラム言語といいます。

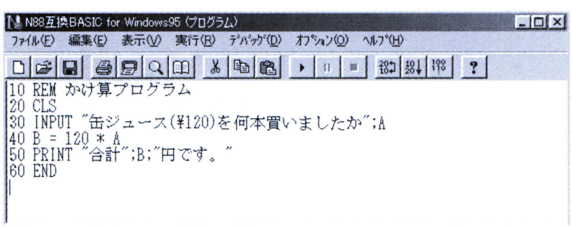
コンピュータは，0か1の2進数で表現される機械語（マシン語）しか理解することができません。しかし，機械語でプログラムをつくることは，人間には直観的にわかりづらく非常に困難な作業になります。そこで人間にも理解しやすい表現を使った，「高級言語」と

呼ばれるものが開発されました。BASIC（Beginner's All purpose Symbolic Instruction Code）言語は，その高級言語のうちの一つです。

プログラミングの第一歩！ということで，さっそくBASICを使ってコンピュータに仕事をさせてみましょう。互換BASICを起動し，次のプログラムを入力してみましょう。プログラムの入力は基本的に半角英数文字で行います。コメント（後述）や"（ダブルクォーテーション）に挟まれた文字列についてのみ，全角文字を使用することができます。

＊サンプルプログラム01

```
10 REM かけ算プログラム
20 CLS
30 INPUT"缶ジュース(¥120)を何本買いましたか";A
40 B = 120＊A
50 PRINT "合計";B;"円です。"
60 END
```



▲プログラムの入力画面

（画面は，N88互換BASIC for Windows95を使用しています。）

プログラムを入力したら早速実行してみましょう。

実行画面に，

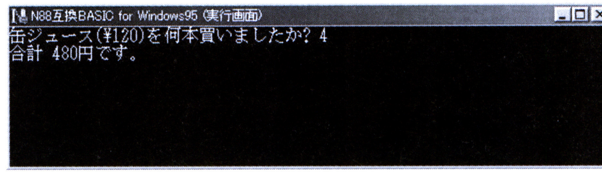
缶ジュース (¥120)を何本買いましたか？

と表示され，カーソルが点滅しますので好きな数字を入力します。

ここでは，4を入力してみます。キーボードから 4 を入力した後 Enter を押してください。プログラムが次に進み，

合計 480円です。

と表示されプログラムは終了します。



▲プログラムの実行画面

プログラム解説

単純なプログラムですが，たくさんのBASICの命令語が使われています。コンピュータはプログラムの各行の先頭に書かれている数字の小さい順に命令を受け取り処理していきます。この数字のことを**"行番号"**といいます。これはBASIC言語独特の表現です。10ずつ番号を付けているのは，後でプログラムを追加する際に容易になるからです。

プログラム（命令）を受け取ったコンピュータはどのような処理をおこなっているのか分析していきましょう。

○10行目の"REM"は，プログラムを見たときにわかりやすくするためのコメントを書くための命令で，この後に好きな文字を書くことができます。プログラムを実行しても何も表示されません。REMの変わりに，'（シングルクォーテーション）を使用することもでき，こちらを使うプログラマーの方が多いです。

10 ' かけ算プログラム

行の先頭だけでなく，他の命令のあとにつなげて書くこともできます。

60 END 'プログラムを終了する

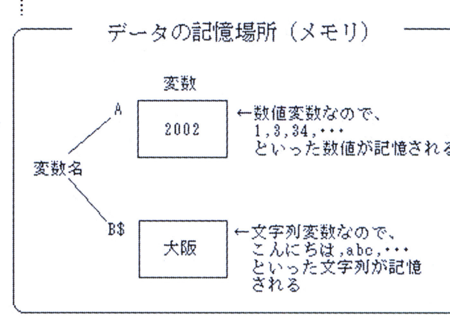
○20行目の"CLS"は，実行画面にこれまで表示されていたものを全て消去する命令です。

○30行目の"INPUT"は，キーボードからの入力を求め**変数**に代入する命令です。

変数とは，コンピュータが扱える2種類のデータ（数値と文字列）の代わりをするもので，数値の代わりをするときは**数値変数**，文字の代わりをするときは**文字列変数**と呼びます。

ここでは，Aという数値変数を使用しています。この変数Aは，プログラムを実行するまでどんな数値が入るかわかりません。INPUT命令で，"缶ジュース (¥120)を何本買いましたか？"と入力を求められることで数値が決定します。このように変数Aは数値の代わりとして使用されます。決定した数値は変数に代入され，コンピュータはその値をメモリ（記憶場所）に記憶します。

変数には英数文字を使い自由に名前（**変数名**）をつけることができます。ただし１文字目は英文字である必要があります。また，文字列変数の場合は変数名の後ろに"$"をつけます。文字列を代入する場合は，文字を"（ダブルクォーテーション）で囲みます。囲まれた文字は，例え数値や式であっても文字列とみなされます。



▲プログラムでは，A=2002，B$="大阪"となります。

○40行目は，数値変数Bに120＊A（数値変数A）の計算結果を代入する命令です。"＊"は，×（かける）を表すもので，120 × A のかけ算を表します。30行目のINPUTでキーボードから求めた値が，4の場合 A = 4 となり，120 × 4 の計算結果がBに代入されB = 480 となります。

A = 4のとき
B = 120＊A =120 × 4 = 480

≪計算式で使える記号（演算記号）≫

+	足し算	7 + 2 = 9
-	引き算	7 - 2 = 5
*	かけ算	7 * 2 = 14
/	わり算	7 / 2 = 3.5
^	べき乗	7 ^ 2 = 49
¥	わり算の商	7 ¥ 2 = 3
MOD	わり算の余り	7 MOD 2 = 1
()	括弧	2 * (7+2) = 18 2(7+2)はエラーになります。

演算の優先順位は，次のようになります。

(高)
^
* /
¥
MOD
+ −
(低)

例）
A = 1 + 2 * 3 - 4 = 1 + 6 - 4 = 3
B = (1 + 2) * (4 MOD 3) = 3 * 1 = 3

C＝A＊B
　このように変数も上記の計算記号を使って計算できます。ただし，A＊N＄のように数値変数と文字列変数を計算することはできません。

M＄＝N＄＋O＄
　文字変数の場合，＋ のみ使うことができます。これは，N＄とO＄の中身を連結しM＄に代入するという意味になります。

```
10 N$ = "ベーシック"
20 O$ = "マガジン"
30 M$ = N$ + O$
40 PRINT M$
```

　とすると，画面には ベーシックマガジン と表示されます。

○50行目の"PRINT"は，"（ダブルクォーテーション）で挟まれた文字列を表示したり，変数の中身を表示する命令です。

・PRINT "合計"
　　合計　と表示します。
・PRINT A
　このときの数値変数Aの中身を表示します。
・PRINT "合計"; A
　文字と変数の間を;（セミコロン）でつなぐと，文字と変数の値を一緒に表示します。
・PRINT　A＋B
　数値変数Aと数値変数Bの値を計算して結果を表示します。
　サンプルプログラムの40行目と50行目は，次のようにまとめることができます。

```
50 PRINT "合計";120＊A;"円です。"
```

　このとき40行目は必要なくなります。変数を一つしか使わないためプログラムがシンプルになりますね。

○60行目の"END"は，プログラムを終了させる命令です。行末の場合は省略できます。
　途中でプログラムを中断させたい時などに使用します。

【ワンポイント】～変数名～

変数名をつけるときは，少し長くなってもわかりやすい名前をつけるほうが，後でプログラムを読んだときにわかりやすくなります。サンプルプログラム01の場合，変数A を 変数HONSUUに，変数B を変数GOUKEIにすると後で見たときにより分かりやすくなります。

```
30 INPUT "缶ジュース(¥120)を，何本買いましたか"; HONSUU
40 GOUKEI = 120 * HONSUU
50 PRINT "合計"; GOUKEI; "円です。"
```

（ただし使用するBASICによっては変数名の文字数に制限がある場合がありますので，ご使用のBASICについて確認してください。）

練習問題にチャレンジ！

　わずか6行のプログラムですが内容はとても充実していますね。ここまででBASICの基本30パーセントをマスターしたと言っても過言ではありません。それでは練習問題にチャレンジして，ここまでの復習をしてみましょう！

問１． 行番号とは，プログラムを記述する行の番号です。コンピュータは行番

号の（ｱ　　　）ほうから（ｲ　　　）ほうに実行します。

問 2．（ｳ　　　）命令は，プログラムの説明やコメントなどを記述するもので，この命令の後の文字はプログラムには反映されません。

問 3．CLSは，実行画面に表示されているものを（ｴ　　　）するための命令です。

問 4．INPUTは，（ｵ　　　）からの入力を求め変数に代入する命令です。

問 5．変数Ａ　は，（ｶ　　　）変数といい，変数N＄ は，（ｷ　　　）変数といいます。

変数Aに8を代入する式はA＝8と表し，N＄ に東京を代入する式は（ｸ　　）と表す。

問 6．A＝5のとき，Bの値を求めなさい。

B＝（4＋2）＊A＝（ｹ　　　）

B＝(A MOD 2)＋10／A＝(ｺ　　）

問 7. 次のプログラムを実行すると，実行画面には，（ｻ　　　）と表示される。

```
10 CLS
20 A$ = "今日は"
30 B$ = "いい天気だ"
40 PRINT A$ ; "とっても" +B$
50 END
```

（練習問題の解答は，この講座の最後に掲載。）

プログラムをマスターする

プログラムは，一行一行がコンピュータに対する命令です。コンピュータは人間の命令に忠実に従ってくれます。みなさんは，サンプルプログラム01を一度も間違えずに入力することができましたか？

20 CLA
とか，
30 INPOT

などのようにつづり間違いをして，プログラムがうまく動作せずエラーメッセージが出たことはありませんか？エラーが出るということは人間が正しい命令を出せていないということになります。入力ミスがないか，間違った命令がないか確認しながら，正しくコンピュータに命令を出しましょう。

【ワンポイント】～入力ミス～

互換BASICを使っている初心者にありがちなミスは，プログラムに全角文字を使用してしまうことです。全角のスペースは特に気がつきにくいので注意しましょう。全角文字は，REM命令の後につづくコメントになる文字列や，"（ダブルクォーテーション）で囲まれた文字列の間でしか使用できません。

プログラミングは作文のようなもの。同じ言葉を使って同じ内容を伝えようとしても十人十色で表現が微妙に違いますよね。プログラムも同じ結果が出るものであってもプログラマーによってそれぞれ微妙に違うプログラムになります。覚えた命令語をどういう手順でコンピュータに伝えるのか。これをマスターすることがプログラムをマスターすることになります。

一番の近道はプログラムに慣れること。実際にプログラムを組んだり入力したりして，どういう風にコンピュータが動くのか，どんな処理をするのかということを体験してください。

サンプルプログラムを幾つか紹介します。それぞれ入力・実行してみましょう。そして，プログラムを一行ごとに英語を訳していくかのように読んでいってみてください。慣れてくれば，プログラムを実行しなくても，どういう内容なのかわかるようになってきます。

＊サンプルプログラム02

```
10 CLS
20 INPUT "パソコンを誉めてください。"; A$
30 PRINT
40 PRINT "パソコン>あなたも"; A$; "ですよ"
50 END
```

「30行目のPRINTの使い方に注目」

＊サンプルプログラム03

```
10 CLS : PRINT "数字を2つ入力してください。"
30 INPUT "1つ目の数字を入力してください";A
40 INPUT "2つ目の数字を入力してください";B
50 PRINT
60 PRINT "A+B="; A+B
70 PRINT "A-B="; A-B
80 PRINT "A*B="; A*B
90 PRINT "A/B="; A / B
100 END
```

「10行目に注目」

サンプルプログラム解説

サンプルプログラム02

30 PRINT

PRINTの後に何も指定していません。この場合，何も表示せずに一行改行します。

PRINT "" としても同じ結果になります。

サンプルプログラム03

10 CLS：PRINT "数字を2つ入力してください。"

一行に2つの命令語が入っています。:（コロン）を使うことで，一行に複数の命令を書くことができます。

次回予告

プログラムに欠かすことができない繰り返し処理と分岐処理について解説します。

（練習問題　解答）

ｱ→小さい（少ない）
ｲ→大きい（多い）
ｳ→REM（'）
ｴ→消去（クリア）
ｵ→キーボード
ｶ→数値
ｷ→文字
ｸ→N＄="東京"
ｹ→30
ｺ→3
ｻ→今日はとってもいい天気だ

ゲームを作ってみよう！

舞台は，星音(ほしね)高校パソコン研究部。４人の部員たちが初めてのゲーム制作にチャレンジ！
彼らも「ベーシックプログラミング教室」を読みゲームを作ろう！！と意欲満々！！
はたしてどんなゲームができあがるか！お楽しみに！

登場人物

「ちぃさん」(ちぃ)　・・・２年生。部長。パソコンは詳しいけど，プログラミングははじめて。
「みぃちゃん」(みぃ)・・・２年生。唯一の女子。その大胆な発想にみんないつも驚き。
「よっさん」(よっ)　・・・３年生。頼りになる先輩。とてもやさしいアニキです。自称ゲーマー。
「ゆうちゃん」(ゆう)・・・１年生。パソコンにはあまり詳しくないけど，音楽ならまかせて。

～星音高校，パソコン研究部部室～

みぃ：「もうすぐ文化祭だね～。みんなインターネットで遊んでばかりいないで，何かゲームでも作らない？」
よっ：「やっぱりネットゲームはやめらへんなぁ。こう，奥が深いというかなんというか・・・」
ゆう：「みぃちゃん，それいいアイデアですね。やりましょうよ。部長どう思いますか？」
ちぃ：「ちょうどベーマガで，ベーシックプログラミング教室も始まったみたいだし，パソコン研究部というからには，オリジナルゲームの一つでも作ってみんなをアッと言わせたいなぁ。よし！やってみよう！」
みぃ：「やったー！そうと決まればさっそく作りましょう！！えっとじゃあ早速プログラムを・・・」
ちぃ：「ちょっと待った！いきなりゲームなんて作れないよ。みんな集合！！」

ということで，４人のゲーム制作が始まった。

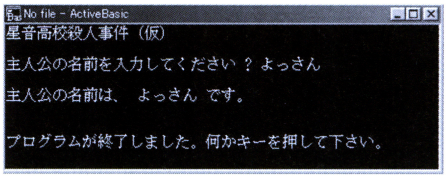
ちぃ：「いきなりゲームを作ろうにも何から始めていいかわからないだろ。ここは，ゲームに一番詳しい自称ゲーマーのよっさん先輩に話を進めてもらおうよ。従兄弟がゲーム制作会社に勤めてるって話だし。」
よっ：「俺？ゲームは遊ぶほうがいいんやけどなぁ・・・ＯＫ！じゃあ，ゲーム制作について説明するからみんなしっかり聞くように。まず，ゲームを作るにはおおまかに次の四つの要素が必要やねん。

・企画（ストーリーやゲームの仕様）
・デザイン(キャラクターや背景などのデザイン)
・サウンド(効果音やBGM)
・プログラム

全部1人で担当して，それぞれゲームをつくるのもいいんだけど，やっぱりみんな得手不得手もあるし，１人で全部となると大変なので担当を決めて４人で一つのゲームをつくることにしよう。まず企画担当は，みぃちゃん。デザイン担当は，俺。サウンド担当は，ゆうちゃん。メインプログラム担当は，ちぃさんでいこう。」
ちぃ：「なるほど。みんなの個性を活かした分担やね。」
みぃ：「じゃあ私はストーリーを考えるね。あとみんな任したよ！」
よっ：「あかんあかん。自分の担当のことだけでなく，他の担当のことも良く知っておかないとあかんで。」
ちぃ：「そうそう。プログラマーのできることや，デザイナーのできることを理解して，それを元にストーリーを作らないと，ストーリーが出来上がってもそんなのゲームにできないってことになってしまうよ。」
みぃ：「そっか。」
よっ：「どんなゲームにするかは4人で考えよう。みんなどんなゲーム作りたい？」
ゆう：「やっぱりドラクエみたいなロールプレイングゲームかな。」
みぃ：「推理小説が好きだから推理アドベンチャーゲームがいいなぁ。」
よっ：「6回しかないねんからそんな大作はあかんよ。プログラム担当のちぃさんはどう思う。」
ちぃ：「そうだね。僕もベーシックプログラミング教室で勉強をはじめたばかりだから，いきなり難しいことできないし，シンプルだけど楽しい！って思えるようなゲームがいいかな。」
みぃ：「両方ミックスしてしまおうよ。RPG風推理アドベンチャーゲームってどう？」
ゆう：「あ，それいいかも！」
ちぃ：「よし決定！！じゃあ，みぃちゃんは次回までにストーリーを考えてね。他のメンバーと相談しながらゲーム制作に無理のないおもしろいストーリーを考えてね。」
みぃ：「ラジャー！！」
ちぃ：「よし，じゃあ僕は早速今回プログラミング教室で覚えた命令を使って，できるところから作っていくね。」
よっ：「今回覚えたところだけやと何ができる？」
ちぃ：「ちょっとやってみるよ。」
～10分後～
ちぃ：「できた！！」
(プログラムVersion0.01)

```
10 '星音高校　パソコン研究部
20 '　　プログラム制作開始　2002/3/1
30 '　　　Version 0.01　2002/3/1
100 CLS
110 PRINT "星音高校殺人事件（仮）"
120 PRINT ""
130 INPUT "主人公の名前を入力してください";PLAYER$
140 PRINT ""
150 PRONT "主人公の名前は，";PLAYER$;"です。"
```

よっ：「早速実行してみよう！ん？“星音高校殺人事件”ってなんや。」
ちぃ：「それは，まだタイトルが決まっていないから仮で書いてるだけ。」
よっ：「そっか。じゃあ続きを…主人公の名前を入力するんやね。じゃあ，“よっさん” [Enter] と…。あ！エラーでたで。あかんなぁ，ちゃんとバグチェックしとかなあかんで。」
ちぃ：「え？エラー！バグチェックって何？」
よっ：「バグズ・ライフっていうディズニーの映画あったやろ。“バグ”というのは虫(Bug)のことで，プログラム中にある間違いのことをゆうねん。『バグがある』というのは，プログラムの中に悪い虫がいる！間違いがある！ということ。バグチェックというのは，プログラムを作った後，バグが潜んでいないかチェックをすることなんや。」
ちぃ：「なるほど。そういえば，プログラム組んで一度も実行しなかったなぁ。ちゃんとバグチェックをしないといけないってことだね。」
よっ：「そうそう。じゃあ，このプログラムからバグを取って。」
ちぃ：「はい。えっと…あ，150行目 にスペルミスを発見！PRONTじゃなくて，PRINTだ！！先輩これでどうですか？」
よっ：「よし。じゃあ，“よっさん” [Enter] と…。お！今度はうまくいった！」



▲ Active Basic の実行画面

よっ：「もう終わり？」
ちぃ：「今回のプログラミング教室で習ったのこれだけなんで…」
よっ：「とてもゲームと呼べないなぁ，これじゃぁ。」
ちぃ：「まだはじまったばかりだし，次回のプログラミング教室がとても楽しみ。」

こうして４人のゲーム制作はスタートした。さてどんなゲームができあがるのか！
次回をお楽しみに！！

この講座で使用しているプログラムデータは，下記のホームページでダウンロードすることができます。
http://www.beeplus.or.jp/bm/basic/

☆BM基礎講座「ベーシックプログラミング教室」
制作　スタジオ スターノーツ
協力　有限会社 ビープラス
(http://www.beeplus.or.jp/)



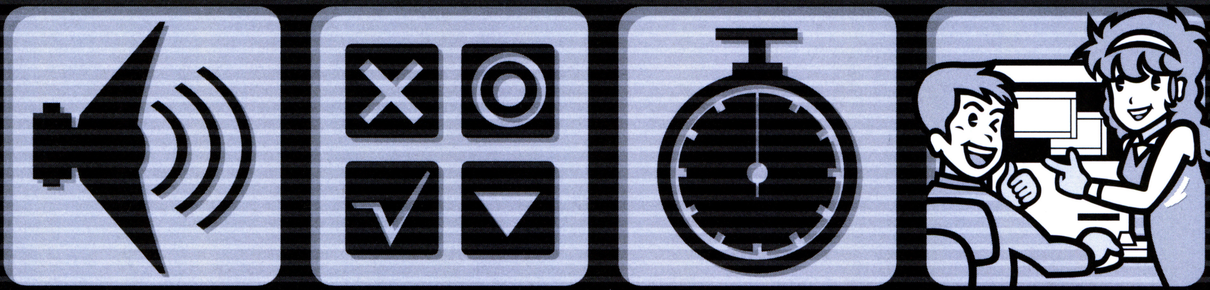
本コーナーは，さまざまな言語に対応したプログラミング学習と創作をテーマに，読者からの投稿作品で成り立っています。これから，プログラミングを勉強しようというみなさんや，プロのプログラマー／クリエイターを目指そうというみなさんにとって役立つ情報が満載です。手近にあるパソコンに入力して，お楽しみください。

それぞれの記事にはプログラムを入力／実行するときの注意点やプログラムの内容，改造方法などが紹介されていますので，よく読んでください。また，“CHECKER FLAG”というコラムがあり，デバッグ博士ことDr.Dがそれぞれのプログラムに対し批評を加えています。口が悪い人なので，かなりキビシイことも言っていますが，自分でプログラムを作ったり，これから投稿しようとしている人にとっては，とても参考になるでしょう。

- Visual Basic
- Visual C++
- Delphi
- F-BASIC
- HSP
- iアプリ
- Wonder Witch

BM特選 投稿プログラムコーナー

プログラマー＆クリエイターへの登竜門



プログラム投稿者へのお願い

自分の作ったプログラムを投稿しようという人はつぎのきまりを守って応募してください。

①プログラム（作品）ファイル ②プログラムの説明，住所・氏名・年齢・職業（学校名）・電話番号（連絡のつく時間も）を書いた原稿のテキスト・ファイルをフロッピーディスクなどにセーブし（プリンタで打ち出した原稿も同封のこと），お送りください。

プログラムの説明には，①内容 ②使いかた・遊びかた ③操作方法 ④プログラムのしくみ（変数表やフローチャートも）⑤プログラムの具体的入力方法，注意点 ⑥改造のアドバイス，作ったとき苦労した点 ⑦使用機種名 ⑧言語（バージョン，モードなども）などを書きます。

注1）プログラムは必ずチェックし，ミスを防ぐために同じものを2〜3入れてもらえば安心です。

注2）プログラムの説明は，テキスト・ファイルをフロッピーディスクなどにセーブし，念のためプリンタで打ち出したものも同封することが基本ですが，手書き原稿でも受け付けます。その際，できるだけていねいに，マス目のある用紙を使用してください。

注3）ディスクのラベル，封筒の裏面にもプログラム名（ファイル名も），原稿ファイル名，住所，氏名，電話番号を必ず記入してください。

内容はオリジナルのもので，未発表のものに限ります。もし，参考にした記事や本がありましたら，書名，号数，記事名，筆者名，記事のタイトルなども明記しておいてください。

お送りいただいたディスクはお返しできませんので，あらかじめバックアップをとっておいてください。

掲載されたプログラムに対しては，規定の原稿料をお支払いし，掲載誌をお送りいたします。

編集部

★本誌とじ込みの愛読者カードについている応募カードも忘れずに同封してください。

あて先：〒141-8715 東京都品川区東五反田1-11-15 電波新聞社マイコンBASICマガジン「特選プログラム・コーナー」係

掲載プログラムのダウンロード／投稿はこちら

http://www.basicmagazine.net/

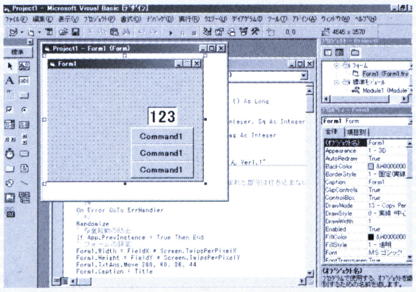
掲載[投稿プログラム]言語

入手方法と入力手順ガイド

Visual Basic

製品紹介

Visual Basic(以下VB)とは，マイクロソフトから発売されている，Windows上で動作するプログラム作成ツールです(開発画面は**写真1**)。現在はVersion6.0が最新版となっています。また，EXE（実行ファイル）作成などの機能を制限した評価版(体験版)も存在します。



▲写真1

入手方法

製品版は，各種類ともパソコンショップなどで購入可能です。学生・学校職員向けに価格を抑えたアカデミックパッケージも用意されています。

また，評価版は本誌で連載していた記事をまとめたムック本『入門Visual Basic　6.0/5.0講座』(税込み2100円)をはじめ，いくつかの書籍に付録として収録されています。

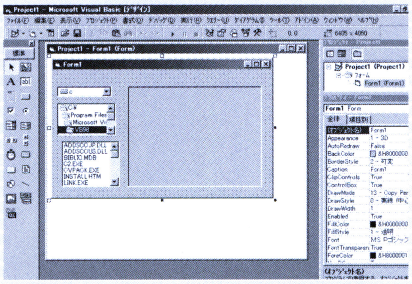
本誌プログラムの基本入力方法

①VBを起動後，標準EXEを選択します(写真2)。

②掲載プログラムのフォームの配置図を



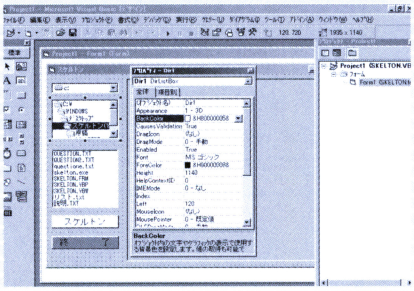
▲写真2



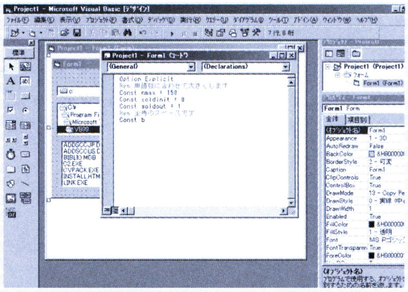
▲写真3

参考にしながら，オブジェクトをフォーム上に配置します(写真3)。

③掲載しているプロパティ表にそって，配置したオブジェクトのプロパティを設定（すでに入っている値を変更）していきます(写真4)。プロパティによっては，別に作成した画像ファイルなどを指定す



▲写真4



▲写真5

る場合がありますので，その場合は本誌に掲載されている画像写真を参考に，Windows付属の「ペイント」などのソフトを用いて画像を作成します。

④フォームのコードウィンドウを表示し，掲載されているプログラム・リストを入力していきます(写真5)。モジュールのリストがある場合は，同様にモジュールのコードウィンドウからリストを入力していきます。

⑤フォームが複数ある場合は，そのフォームの数だけ，②〜④の作業を繰り返します。

⑥すべての入力作業が終了したら，実行ボタンを押しプログラムを実行させます(写真6)。

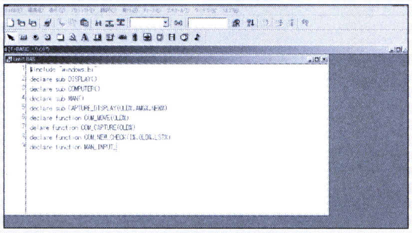


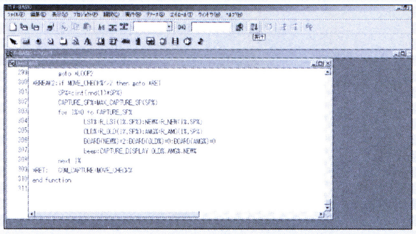
▲写真6

F-BASIC(Windows版)

製品紹介

F-BASICとは，富士通ミドルウェアから発売されているWindows用プログラム言語です。現在はVersion6.3が最新版となっています。


▲写真1


▲写真2

入手方法

パソコンショップなどで購入可能です。

本誌プログラムの基本入力方法

①F-BASICを起動後，画面上のエディタから，掲載されているリストを入力します(写真1)。

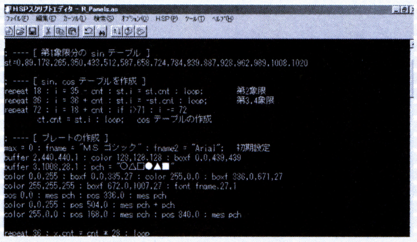
プログラムによっては，別途画像ファイルを必要とする場合があります。その場合は，誌面に掲載されている画像写真を参考に，Windows付属の「ペイント」などのソフトを用いて画像ファイルを作成します。

②リストを入力後，実行ボタンを押してプログラムを実行させます(写真2)。

HSP

製品紹介

HSPとは，フリーで使用できるWindows用プログラム言語です(開発画面は写真1)。現在はVersion2.55が最新版となっています。


▲写真1

入手方法

HSPは，以下のホームページからダウンロードすることができます(写真2)。

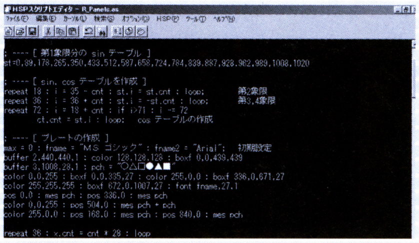
http://www.onionsoft.net/hsp/

また，パソコン雑誌やHSP関連の書籍などの付録として収録されていることもあります。

本誌プログラムの基本入力方法

①HSPを起動後，画面上のエディタから，掲載されているリストを入力します。

プログラムによっては，別途画像ファイルを必要とする場合があります。その


▲写真2

場合は，誌面に掲載されている画像写真を参考に，Windows付属の「ペイント」などのソフトを用いて画像ファイルを作成します。

②リストを入力後，実行ボタンを押してプログラムを実行させます。

iアプリ

製品紹介

iモード対応の携帯電話を使用し実行します。プログラミングはパソコンで行います。

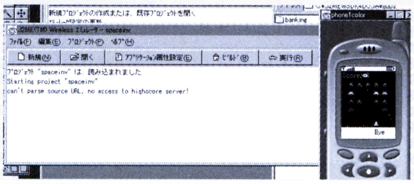
入手方法

携帯電話ショップで購入可能です。

本誌プログラムの基本入力方法

①サンマイクロシステムズのWEBサイト（http://java.sun.com/）からJava2の開発環境「Java 2 SDK Standard Edition 1.3.0」をダウンロードします。インストールすると“C:¥jdk1.3”というディレクトリにJavaコンパイラが展開されます。

② NTTドコモのWEBサイト（http://www.nttdocomo.co.jp/i/java.html）からiアプリの開発ツール「J2ME Wireless SDK for the DoJa release2.2（以下，DoJa)」をダウンロードします。ダウンロードには名前の登録が必要です。インストールすると“C:¥J2MEWSDK4DOJA”フォルダに主要なファイルが展開されます。


▲写真1

③スタートメニューから「J2ME～DoJa」を選択して，「KToolBar」を起動します(写真1)。

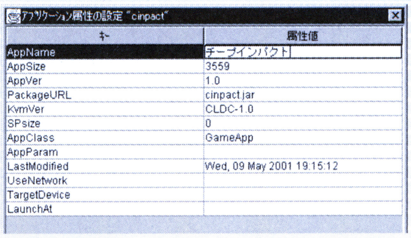
④「新規」ボタンを押して，新規プロジェクト名を入力すると，“C:¥J2MEWSDK4DOJA¥apps”フォルダの下にプロジェクト名のフォルダと,その下に“bin”,“res”,“src”フォルダが作成されます。

DoJaは，テキストエディタを搭載していないので，別途メモ帳などのテキストエディタソフトでソースファイルを作成します。入力方法は，誌面に掲載されているリストを入力し，指定されたファイル名で保存します。ファイル名を間違えると正しくビルドされないことがあります。また，DoJaの場合，jamファイルは

次のページにつづく

ソフト側で作成できるため，入力する必要はありません。

プログラムによっては，別途画像ファイルを必要とする場合があります。その場合は，誌面に掲載されている画像写真を参考に，Windows付属の「ペイント」などのソフトを用いて画像ファイルを作成します。



▲写真2

⑤“res”フォルダに画像（gif）や音楽ファイル（mld），“src”フォルダにはソースファイル（java）を格納します。

⑥「アプリケーション属性設定」ボタンを押して，キーの値を入力します(写真2)。(“AppSize”と“LastModified”は自動的に更新されます)。

AppName = (アプリケーション名)

AppVer = 1.0

AppClass = (最初に実行するクラス名)

SPsize = (スクラッチパッドのサイズ)

KvmVer = CLDC-1.0

⑦「ビルド」ボタンを押すと，コンパイルとプリベリファイが一括して行われ，jarファイルとjamファイルが“bin”フォルダに生成されます。jarファイルは“res”フォルダの中身とクラスファイルがzip形式で圧縮されたものです。

⑧エディタで“C:¥J2MEWSDK4DOJA¥lib¥devices¥phone1color”フォルダの“phone1color.properties”を以下のように書き換えます（値は120×130ピクセル時の設定です）。書き換えたら「編集ー設定」をメニュー選択して，「デバイス指定」を“phone1color”にします。

```
:
screen.x=27
screen.y=65
:
screen.width=120
screen.height=156
:
icon.up: 44, 138, off
:
icon.down: 44, 146, off
:
screenPaintableRegion.width=120
screenPaintableRegion.height=130
:
softbutton.0=0,138,36,18, softButton
softbutton.1=90,138,36,18, softButton
:
```


▲写真3

⑨「実行」ボタンを押すと，携帯電話エミュレータ上でｉアプリが実行されます(写真3)。

Visual C++

製品紹介

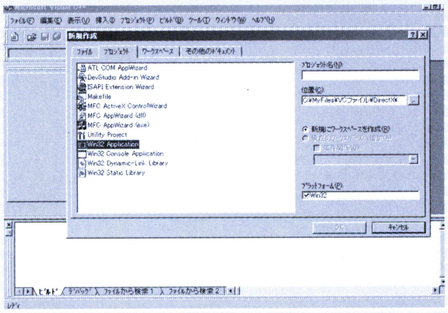
Visual C++(以下VC)とは，マイクロソフトから発売されている，Windows上で動作するC言語によるプログラム作成ツールです。現在はVersion6.0が最新版となっています。

入手方法

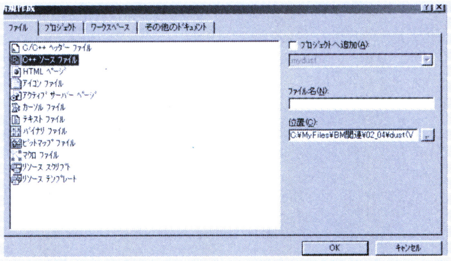
パソコンショップなどで購入可能です。学生・学校職員向けに価格を抑えたアカデミックパッケージも用意されています。

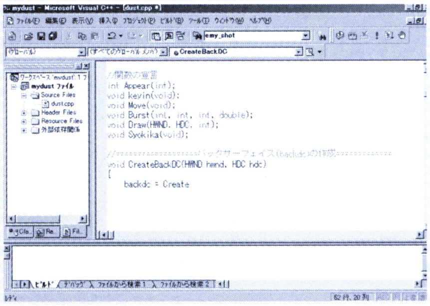
本誌プログラムの基本入力方法

①VC++を起動後，メニューからファイル-新規作成を選択し，本誌の説明で指定されているプロジェクトを選択します（写真１）。


▲写真1

②プロジェクト作成後，同じくファイル-新規作成から，作成するファイルの種類を選択し，掲載されているプログラム・リストを入力します(写真２，３)。

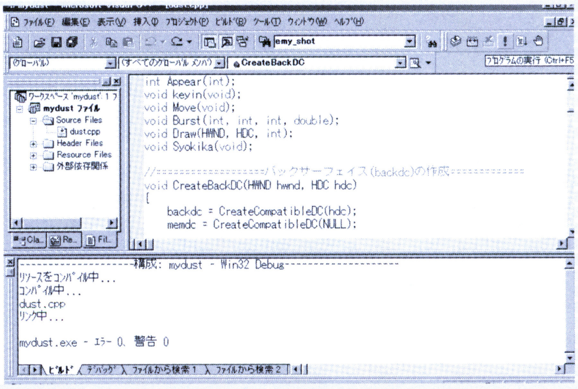

▲写真2


▲写真3

※プログラムによっては，別途画像ファイルを必要とする場合があります。その場合は，誌面に掲載されている画像写真を参考に，Windows付属の「ペイント」などのソフトを用いて画像ファイルを作成します。

③すべての入力作業が終了したら，ビルドボタンを押しプログラムのビルドを行います。ビルド後に，実行ボタンでプログラムを実行させます(写真４)。

※VC++の場合，プログラムの作成方法には多くの形態があり，掲載されるプログラムによってはフォームの作成などを行う場合もあります。その際には，掲載されている説明文を参考にプログラム入力を行ってください。


▲写真4

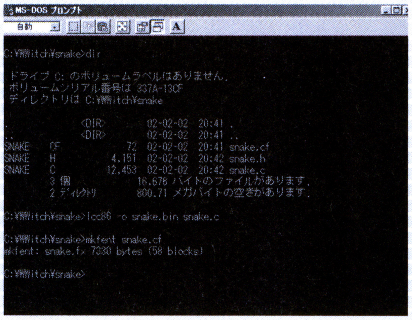
WonderWitch

製品紹介

WonderWitchとは，バンダイの携帯ゲーム機「ワンダースワン」および「ワンダースワンカラー」用のアプリケーションが制作できるプログラミングツールです。パソコンで制作したプログラムの実行は，ファイルを専用カートリッジへ転送して，「ワンダースワン」および「ワンダースワンカラー」上で行います。

入手方法

基本的には，インターネット上での通信販売ですが，一部の店舗では店頭販売を行っています。詳しい購入案内は，WonderSwanWeb内のワンダーウィッチのページhttp://www.swan.channel.or.jp/wonderwitch/を参照してください。

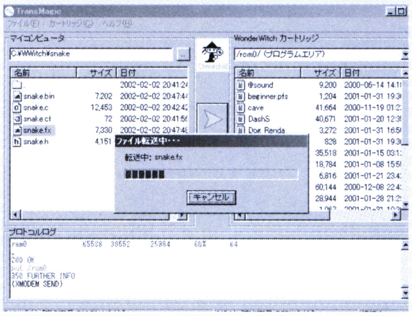


▲写真1

本誌プログラムの基本入力方法

①本誌に掲載されているリストを，メモ帳などのエディタで入力し，ファイルを保存します。

②入力したC言語のファイル(拡張子が.c)をWonderWitchに付属しているコンパイラでコンパイルします。コンパイル後，情報ファイル(拡張子が.cf)に対してバイナリ変換ツールを使用し，プログラムファイル(*.fx) を生成します(写真1)。



▲写真2

③生成したプログラムファイルを，WonderWitchに付属しているファイル転送ツール「TransMagic」を使って，専用カートリッジに転送します(写真2)。

④転送したプログラムを，専用カーリッジのランチャ画面から起動します。

Delphi

製品紹介

Delphiとは，ボーランドから発売されている Windows上で動作するPascal言語によるプログラム作成ツールです。現在はVersion6が最新版となっています。

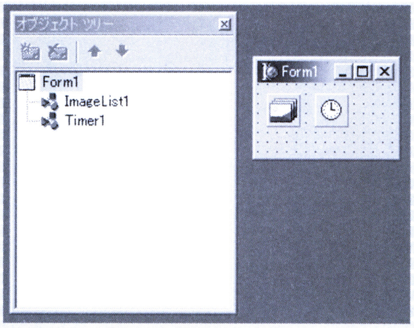
入手方法

パソコンショップなどで購入可能です。学生・学校職員向けに価格を抑えたアカデミックパッケージも用意されています。また，インターネット上からダウンロードできる無償バージョンも用意されています。下記のURLからダウンロードが可能です。

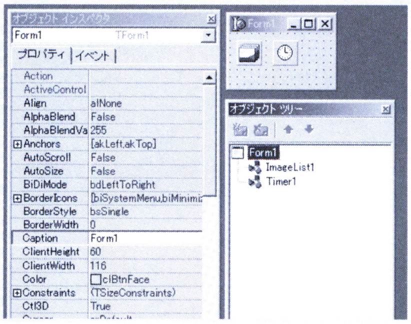
http://www.borland.co.jp/delphi/personal/

本誌プログラムの基本入力方法

①Delphiを起動後，掲載プログラムのフォームの配置図を参考にしながら，オブジェクトをフォーム上に配置します(写真1)。

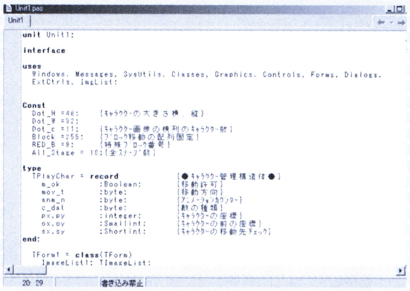


▲写真1



▲写真2

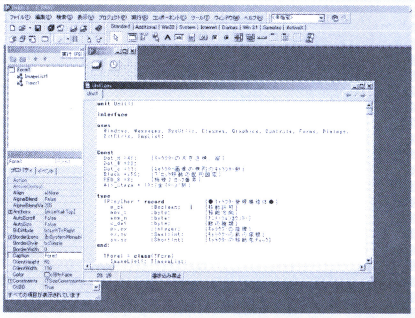
②掲載しているプロパティ表にそって，配置したオブジェクトのプロパティを設定していきます(写真2)。プロパティによっては，別に作成した画像ファイルなどを指定する場合がありますので，その場合は本誌に掲載されている画像写真を参考に，Windows付属の「ペイント」などのソフトを用いて画像を作成します。



▲写真3

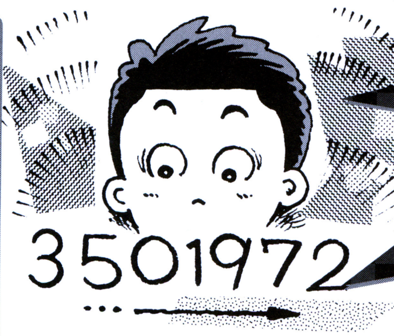
③フォームのコードウィンドウを表示し，掲載されているプログラム・リストを入力します(写真3)。

④すべての入力作業が終了したら，実行ボタンを押しプログラムを実行させます(写真4)。



▲写真4

Windows(Visual Basic V6.0)用



スクロールする数字を記憶せよ

スピードスター

フリーランス（27歳）

はじめに

BASICマガジンは10歳～17歳くらいの間，ずっと読んでました。先日，何気なく本屋で見つけ懐かしく思い，読んでみるとホームページで投稿プログラムなどがダウンロード出来るとの事で早速拝見。さらにメールで投稿できる事を知り，早速の投稿を試みました。

内容

動体視力と記憶力がモノを言うゲームです。画面を流れる数字を覚え，答えていくゲームです。正解する度に流れる数字の長さが長くなったり，スクロールスピードがアップしたりします。

操作方法

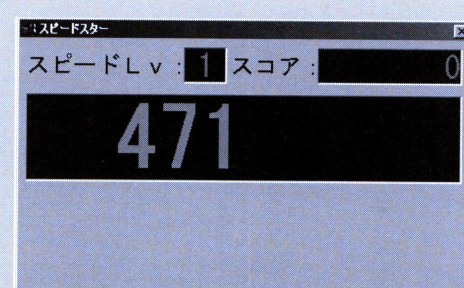
①**プログラム**を実行します。
②画面にあるSTARTボタンを押します**(写真１)**。
③カウントダウン後に,ある桁数の数字が流れます**(写真２)**。この流れる数字を覚えます。
④画面に０～９までの数字が現れますので，覚えた数字を入力します**(写真３)**。入力方法には，数字をクリックする方法と，キーボードで入力する方法の２とおりあります。
⑤数字を入力後，SELECTボタンを押します。キーボード入力の[Enter]キーでもOKです。
⑥正解の場合②～⑤を繰り返します。不正解の場合は，ゲーム終了です**(写真４)**。
⑦画面にあるEXITボタンで終了します。

改造のアドバイス

・回答に時間制限を設けるのも面白い



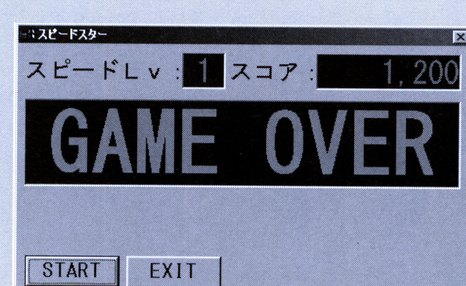
《写真１》移動してくるする数字を暗記するゲーム



《写真２》この画面では“471”が，右側から移動してくる

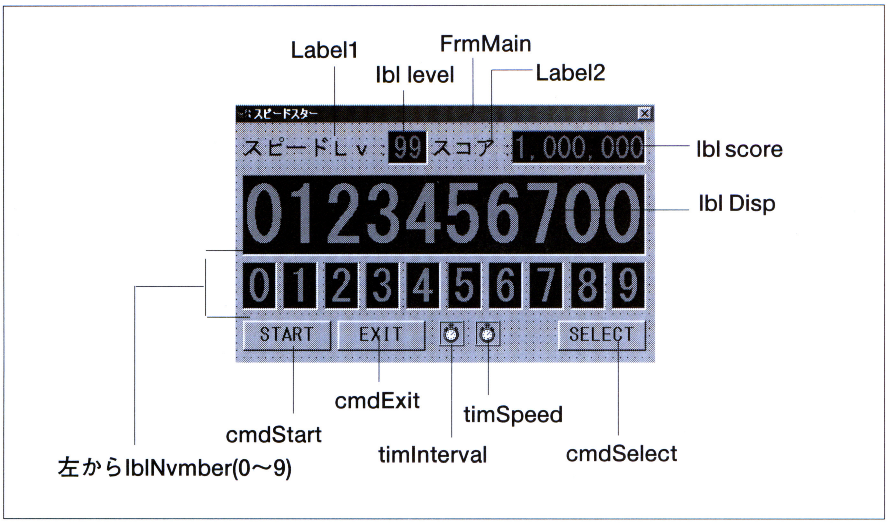


《写真３》覚えた数字を画面に入力。正解なら次へ進む



《写真４》一度でも間違えるとゲームオーバー

《OVER FLOW》進・入学シーズン到来！みなさんの中にも，これからはじまる新しい生活に期待と希望で胸を膨らませている人も多いことでしょう。本誌も多くの読者の期待に応えるべく，今月から新企画も盛り込みました。ご意見・ご感想お聞かせください。



《第1図》フォームの配置図

と思います。

・制限時間で点数の計算方法を変えるのもアリでしょう。

・ラベルコンポーネントでは表示系統が弱いので，他のコンポーネントでも試してみてください。

・スピードと長さはプログラム上部の定数“Const”宣言部で変わります。

CHECKER FLAG

Dr.D：比較的単純な作りのゲームなので，リストも難しい命令や処理などがなく単純に作られておる。VBでのゲームプログラムの入門にはもってこいのリストかもしれんな。ゲーム面で肝となる問題の数字のスクロールを，ピクチャボックスのCaption指定だけで行っておるのが，特徴ではあるのじゃが…。

つぐ美：そのスクロールがちょっと見えにくいのよね。スピードが上がると，そのスクロールスピードでなく，描画時の画面のちらつきによって数字が見えにくくなるわ。この部分でもう一工夫を！

影：あと，正解のときに何か演出がほしいなぁ。今のままだと何も表示がないから，正解しても実感がない。こういうチョットした演出面も，ゲームプログラムでは大事になってくるゾ。

入力方法は30ページをご覧ください。

《第1表》プロパティ表

フォームの設定	
Name	frmMain
BorderStyle	1 '固定(実線)
Caption	スピードスター
Height	4290
Left	0
Icon	(ｱｲｺﾝ)
KeyPreview	-1 'True
LinkTopic	Form1
MaxButton	0 'False
MinButton	0 'False
ScaleHeight	3915
ScaleWidth	6930
StartUpPosition	2 '画面の中央
Top	0
Width	7020
コマンドボタンの設定	
Name	cmdSelect, cmdExit, cmdStart
Font	ＭＳ ゴシック
…Size	16
…ｽﾀｲﾙ	太字
Height	495
Top	3240
Visible	0 'False
Width	1455
Name	cmdSelect
Caption	SELECT
Left	5340
TabIndex	5
Name	cmdExit
Caption	EXIT
Left	1680
TabIndex	4
Name	cmdStart
Caption	START
Left	120
TabIndex	3
ラベルの設定	
Name	lblDisp
Alignment	2 '中央揃え
BackColor	&H00000000&
BorderStyle	1 '実線
Caption	123456700
Font	ＭＳ ゴシック
…Size	64
…ｽﾀｲﾙ	太字
ForeColor	&H00C0C000&
Height	1335
Left	105
TabIndex	17
Top	840
Width	6690
Name	Label1
Alignment	1 '右揃え
BackColor	&H00FFFFFF&
BackStyle	0 '透明
Caption	スピードＬｖ：
Font	ＭＳ ゴシック
…Size	18
Height	360
Left	105
TabIndex	16
Top	217
Width	2520
Name	Label2
Alignment	1 '右揃え
BackColor	&H00FFFFFF&
BackStyle	0 '透明
Caption	スコア：
Font	ＭＳ ゴシック
…Size	18
Height	360
Left	3240
TabIndex	0
Top	217
Width	1440
Name	lblLevel
Alignment	2 '中央揃え
BackColor	&H00000000&
BorderStyle	1 '実線
Caption	99
Font	ＭＳ ゴシック
…Size	24
ForeColor	&H000000FF&
Height	555
Left	2520
TabIndex	2
Top	120
Width	660
Name	lblScore
Alignment	1 '右揃え
BackColor	&H00000000&
BorderStyle	1 '実線
Caption	"1,000,000"
Font	ＭＳ ゴシック
…Size	24
ForeColor	&H000000FF&
Height	555
Left	4575
TabIndex	1
Top	120
Width	2220
Name	lblNumber(0〜9)
Alignment	2 '中央揃え
AutoSize	-1 'True
BackColor	&H00000000&
BorderStyle	1 '実線
Font	ＭＳ ゴシック
…Size	36
…ｽﾀｲﾙ	太字
ForeColor	&H00C0C000&
Height	780
Top	2280
Visible	0 'False
Width	570
Name	lblNumber(0)
Caption	0
Index	0
Left	105
TabIndex	15
Name	lblNumber(1)
Caption	1
Index	1
Left	785
TabIndex	6
Name	lblNumber(2)
Caption	2
Index	2
Left	1465
TabIndex	7
Name	lblNumber(3)
Caption	3
Index	3
Left	2145
TabIndex	8
Name	lblNumber(4)
Caption	4
Index	4
Left	2825
TabIndex	9
Name	lblNumber(5)
Caption	5
Index	5
Left	3505
TabIndex	10
Name	lblNumber(6)
Caption	6
Index	6
Left	4185
TabIndex	11
Name	lblNumber(7)
Caption	7
Index	7
Left	4865
TabIndex	12
Name	lblNumber(8)
Caption	8
Index	8
Left	5545
TabIndex	13
Name	lblNumber(9)
Caption	9
Index	9
Left	6225
TabIndex	14
タイマーの設定	
Name	timInterval
Enabled	0 'False
Interval	300
Left	3360
Top	3240
Name	timSpeed
Enabled	0 'False
Interval	201
Left	3960
Top	3240

★デジタル・データは一瞬で消えるからこわいですね。掲示板のログとか。（北海道恵庭市・**本名吉野22歳**）……【**編集長**：さ，会議をはじめるぞ！あれ？影くんは？**編**：さっきまでいたんですけどね。**Dr.D**：一瞬で消えるのは，デジタル・データだけじゃないな。】

《リスト》スピードスターのプログラム・リスト

```
Option Explicit

Const MIN_LEN = 3   '問題の最短長
Const MAX_LEN = 6   '問題の最大長
Const INI_SPD = 211 '初期スピード
Const NXT_SPD = 30  'アップスピード

Dim mstrMondai      As String   '問題
Dim mstrData        As String   '表示データ
Dim mintLen         As Integer  '問題の長さ
Dim mintInterval    As Integer  '待ち時間
Dim mintLevel       As Integer  'レベル
Dim mlngScore       As Long     'スコア

Private Sub Form_Load()

    '２重起動のチェック
    If App.PrevInstance = True Then
        Call cmdExit_Click  '終了
    End If

End Sub

'ゲームスタート
Private Sub cmdStart_Click()

    Call subInit
    Call subInitNum
    Call subCreate
    Call subOnData
    timInterval.Enabled = True 'タイマー起動

End Sub

'初期化
Private Sub subInit()

    mintLen = MIN_LEN
    mintLevel = 1
    mlngScore = 0
    timSpeed.Interval = INI_SPD
    cmdStart.Visible = False
    cmdExit.Visible = False

End Sub

'カウントダウンの初期化
Private Sub subInitNum()

    mintInterval = 3
    lblDisp.Alignment = 2
    lblDisp.Caption = "3"

End Sub

'問題作成
Private Sub subCreate()
    Dim intLoop     As Integer  'ループ用変数
    Dim intRet      As Integer  '乱数用変数

    Randomize           '乱数の初期化
    mstrMondai = ""     '問題の初期化

    For intLoop = 1 To mintLen
        intRet = Int(9 * Rnd)
        mstrMondai = mstrMondai & CStr(intRet)
    Next intLoop

    mstrData = Space$(10) & mstrMondai & Space$(10)

End Sub

'データの表示
Private Sub subOnData()

    lblLevel.Caption = CStr(mintLevel)
    lblScore.Caption = Format(mlngScore, "#,##0")

End Sub

'READY
Private Sub timInterval_Timer()

    mintInterval = mintInterval - 1
    lblDisp.Caption = CStr(mintInterval)
    If mintInterval = 0 Then
        timInterval.Enabled = False
        lblDisp.Caption = ""
        lblDisp.Alignment = 1
        timSpeed.Enabled = True
    End If

End Sub

'問題の表示
Private Sub timSpeed_Timer()

    mstrData = Mid(mstrData, 2)
    lblDisp.Caption = mstrData
    lblDisp.Refresh
    If Len(mstrData) = 10 Then
        timSpeed.Enabled = False
        Call subSwitch(True)
        lblDisp.Caption = ""
    End If

End Sub
```

★職場のPCが急な雨漏りで水をかぶって廃棄処分。もったいないので持ち帰って電源を入れると運良く動いた。ラッキー！NECの初期型バリュースターですが，付属モニタをノートPCにつないで使ってます。（島根県松江市・**NSR250R+横浜ナンバー37歳**）……【**編：…。**】

```
'回答Noの表示非表示
Private Sub subSwitch(bSwitch As Boolean)
    Dim intLoop As Integer  'ループ用変数

    For intLoop = 0 To 9
        lblNumber(intLoop).Visible = bSwitch
    Next intLoop
    cmdSelect.Visible = bSwitch

End Sub

'選ぶ
Private Sub lblNumber_Click(Index As Integer)

    lblDisp.Caption = lblDisp.Caption & CStr(Index)

End Sub

'キーボードで選ぶ
Private Sub Form_KeyPress(KeyAscii As Integer)

    If cmdSelect.Visible = False Then
        Exit Sub
    End If

    If (KeyAscii >= 48) And (KeyAscii <= 57) Then
        lblDisp.Caption = lblDisp.Caption & Chr(KeyAscii)
    End If

End Sub

'答え合わせ&次のステップ
Private Sub cmdSelect_Click()

    Call subSwitch(False)
    If lblDisp.Caption = mstrMondai Then
        mlngScore = mlngScore + fncTokuten
    Else
        Call subGameOver
        Exit Sub
    End If

    Call subNext
    If timSpeed.Interval <= NXT_SPD Then
        subGameClear
        Exit Sub
    End If

    Call subCreate
    Call subOnData
    Call subInitNum
    timInterval.Enabled = True 'タイマー起動
```

```
End Sub

'次の問の設定
Private Sub subNext()
    Dim intInterval As Integer

    mintLen = mintLen + 1
    mintLevel = mintLevel + (mintLen ¥ (MAX_LEN + 1))

    If mintLen = (MAX_LEN + 1) Then
        mintLen = MIN_LEN
        With timSpeed
        intInterval = .Interval
        .Interval = .Interval - NXT_SPD
        End With
    End If

End Sub

'得点の計算
Private Function fncTokuten() As Long

    fncTokuten = mintLevel * 100 * mintLen

End Function

'ゲームクリア
Private Sub subGameClear()

    lblDisp.Alignment = 2
    lblDisp.Caption = "CLEAR!"
    cmdStart.Visible = True
    cmdExit.Visible = True

End Sub

'ゲームオーバー
Private Sub subGameOver()

    lblDisp.Alignment = 2
    lblDisp.Caption = "GAME OVER"
    cmdStart.Visible = True
    cmdExit.Visible = True

End Sub

Private Sub cmdExit_Click()

    End 'おしまい

End Sub
```

★チェッカーフラッグはあるのですが、それ以外にもっと投稿プログラムを具体的に評価するコーナーを作ってほしいです。たとえば、アイデアな面。(愛媛県新居浜市・赤羊旅団:肉眼鏡17歳)……【Dr.D:明日のスター内だけでは物足りんということじゃな。】

Visual Basic 入力方法は30ページをご覧ください。

Windows(Visual Basic V6.0)用



特殊もぐらたたき

Strike Number
（ストライク ナンバー）

山内　喜智（16歳）

内 容

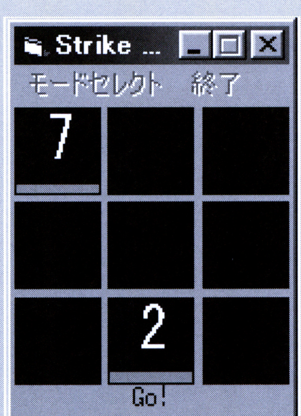
3×3のマスに表示される数字と同じ数字のテンキーを押すことで，そのマスの数字を消していく，もぐらたたき風のゲームです。

同じ数字を押すのであって，表示された場所に対応するキーを押すのではありません。

遊び方

テンキーの1～9キーを使用してゲームをプレイします。

Enterキーを押してゲームをスタートさせると，マスに数字が表示されます(**写真1**)。マスのゲージが満タンになる前に，表示された数字と同じ数字のテンキーを押して，その数字が表示されているマスを消していきます(**写真2**)。ゲージが満タンになるか，表示されていない数字を押すと，Missが+1されます。Missが3を超えると，ゲームオーバーです(**写真3**)。



《写真1》画面に表示される数字を消していく



《写真2》華麗なキー入力が必要なゲーム？



《写真3》すべてのモードでハイスコアに挑戦



《写真4》数字の位置が変わるモードが面白い

各モードについて

ゲーム開始前に，メニューからモードの設定ができます。モードの種類は以下のとおりです。

通常：数字の表示位置がテンキーに対応した位置に表示されます。

ミラー1：通常の左右が逆になります。

ミラー2：通常の上下が逆になります。

ミラー3：通常を180度回転させたものになります(**写真4**)。

ランダム：全消しをしたら，すべての表示位置がランダムに変わります。

通常で遊んでから，別のモードにすると数倍難しくなります。ストループ現象とかいうものを利用しています。

改造のヒント

Const部を変更すれば，簡単な設定の変更が可能です。リスト中のコメントを参考に，お好みで設定してみてください。

スコアUPの“score = score + 1”を“score = score + Fix(time(f)/10)”とすれば，スコアは溜まっているゲージに応じたスコアになります(この場合は，溜まっているほどスコアが多く入る。10で割っているのは，そのままだと得点が多すぎるため)。

★ついに「BASIC STUDIO」とPS2を購入しました！！…でも，どうやって作品を投稿するんだろう。(**福島県郡山市・近内沖尋19歳**)
……【**影**：「BASIC STUDIO」の投稿受け付けると言ってから，その手段にいい案が出ていません。妙案見つかった人連絡ください。】

入力方法は30ページをご覧ください。

苦労した点

表示位置の変更は，最初，変数でしようと思っていましたが，キーを対応させるのが難しくなり，一度作成をやめようかと思いました。しかし，ピクチャボックスの位置を変えるほうが楽だと気付き，今のように変更しました。

CHECKER FLAG

影：一見単純に見えるけど，遊んでみるとけっこう面白い。とくにミラーモードのアイデアが，ゲームをより面白いものにしているゾ。
つぐ美：数字の出現するテンポがいいので，プレイしていて爽快感がありますね。マスを消したときの演出にこだわると,より面白味が増すんじゃないかしら？
Dr.D:意外と見落としがちじゃが，改造のコード記述例や定数へのコメント，フォーム上のシェイプアップなどリスト面でも参考になる作品じゃ。

《リスト》Strike Number のプログラム・リスト

```
Option Explicit

Dim f, ff, fo, fr As Long 'For Next
Dim box(5) As Long '一時格納用
Dim score, miss As Integer
Dim lv, gamelv As Integer
Dim lineupspeed As Integer
Dim time(9) As Long
Dim numflag(9) As Boolean
Dim keyflag(9) As Boolean
Dim gameflag As Boolean
Dim wide, high As Long
Dim x, y As Long '座標一時格納
Dim am, tm As Long 'GetTickCount

'API
Private Declare Function GetTickCount& Lib "Kernel32" ()
Private Declare Function GetKeyState Lib "user32" (ByVal nVirtKey As Long) As Long

'Const
Const MIXFREQUENCY = 25 'mix の 回数
Const MAXMISS = 3 'ミスの最大数
Const FIRSTLINESPEED = 5
Const UPLINESPEED = 1 'ゲージのLvUp時増加量
Const LVUPAVERAGE = 100 'LvUp する Score の 倍数

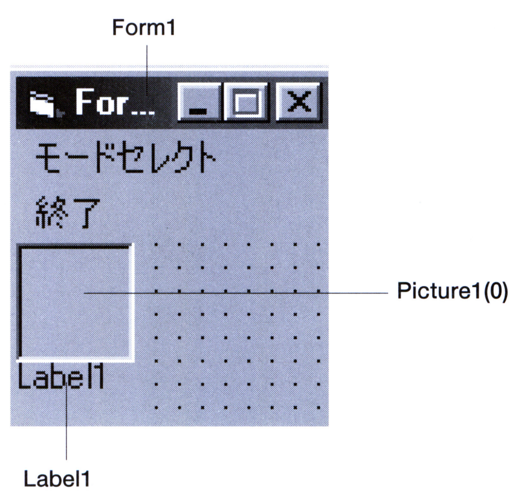
Private Sub Form_Load()

 'Picture1 の初期設定
 Picture1(0).Appearance = 0
 Picture1(0).AutoRedraw = True
 Picture1(0).BackColor = &H0
 Picture1(0).Font = "MS ゴシック"
 Picture1(0).FontSize = 20
 Picture1(0).ForeColor = &HFFFFFF

 'Picture1(1～8) を Load
```

《第1表》プロパティ表

フォームの設定	
Name	Form1
Caption	Form1
Height	1905
Left	105
LinkTopic	Form1
MaxButton	0 'False
ScaleHeight	930
ScaleWidth	1560
StartUpPosition	3 'Windows の既定値
Top	105
Width	1740
ピクチャボックスの設定	
Name	Picture1(0)
Height	615
Index	0
Left	0
ScaleHeight	555
ScaleWidth	555
TabIndex	0
Top	0
Width	615
ラベルの設定	
Name	Label1
Caption	Label1
Height	255
Left	0
TabIndex	1
Top	600
Width	615



《第1図》フォームの配置図

《第2表》メニューの設定

ｷｬﾌﾟｼｮﾝ(P)	名前(M)	ｲﾝﾃﾞｯｸｽ(X)	ﾍﾙﾌﾟｺﾝﾃｷｽﾄ番号(H)	ｼｮｰﾄｶｯﾄ(S)	ﾈｺﾞｼｴｰﾄ位置(O)	ﾁｪｯｸ(C)	有効 (E)	表示(V)	ｳｲﾝﾄﾞｳﾘｽﾄ(W)
モードセレクト	ModeSelect		0	(なし)	0－なし		○	○	
…通常	Mode	0	0	(なし)	0－なし		○	○	
…ミラー1	Mode	1	0	(なし)	0－なし		○	○	
…ミラー2	Mode	2	0	(なし)	0－なし		○	○	
…ミラー3	Mode	3	0	(なし)	0－なし		○	○	
…ランダム	Mode	4	0	(なし)	0－なし		○	○	
終了	Owari		0	(なし)	0－なし		○	○	

★JR田町駅前に「影ラーメン」という店を発見！もしかして…？（東京都世田谷区・PANDA23歳）……【**つぐ美**：誰かさんが歌いだすわよ。**影**：♪もしかして，もしかして～。**つぐ美**：ほら，単純ね。**影**：最近歌う機会も減ったからなー。私の美声を読者にと…。】

★10年ぶりにベーマガを買いました。私が学生のころ，8ビット・パソコンとポケコンにかじりついていて，ベーマガはバイブルでしたが，今も内容が進化していて読み応え十分です。（大阪府吹田市・**あり32歳**）……**【編：**時代とともに歩みつづけるベーマガ，よろしく！**】**

```
For f = 1 To 8
  Load Picture1(f)
  Picture1(f).Left = Fix(f Mod 3) * Picture1(f).Width + Fix(f Mod 3) * 50
  Picture1(f).Top = Fix(f / 3) * Picture1(f).Height + Fix(f / 3) * 50
  Picture1(f).Visible = True
Next f

'ラベル，フォーム の初期設定
Label1.Alignment = 2
Label1.Font = "ＭＳ ゴシック"
Label1.Caption = "Push Enter to Start"
Label1.Top = Picture1(6).Top + Picture1(6).Height
Me.Width = Picture1(8).Left + Picture1(8).Width + Me.Width - Me.ScaleWidth
Me.Height = Label1.Top + Label1.Height + Me.Height - Me.ScaleHeight
Me.Left = (Screen.Width - Me.Width) / 2
Me.Top = (Screen.Height - Me.Height) / 2
high = Me.Height
wide = Me.Width
Me.Caption = "Strike Number"
Me.AutoRedraw = True
Label1.Width = Me.ScaleWidth
Mode(0).Checked = True

Me.Show
Randomize

Picture1(0).Print "Str"
Picture1(1).Print "ike"
Picture1(3).Print "ｽﾄﾗ"
Picture1(4).Print "ｲｸﾅ"
Picture1(5).Print "ﾝﾊﾞ"
Picture1(7).Print "Num"
Picture1(8).Print "ber"

Do
  Call wait(100)
  If (GetKeyState(vbKeyEscape) And &H80) <> 0 Then End
  If gameflag = False And (GetKeyState(vbKeyReturn) And &H80) <> 0 Then
    score = 0
    miss = 0
    gamelv = 1
    lineupspeed = FIRSTLINESPEED
    lv = 2000
    gameflag = True
    Me.Cls
    For f = 0 To 8
      Picture1(f).Cls
      time(f) = 0
    Next f
    ModeSelect.Enabled = False
    Owari.Enabled = False
    Label1.Caption = "Ready..."
    Call Mix
    Call wait(3000)
    Label1.Caption = "Go!"
  End If
  While gameflag = True
    If (GetKeyState(vbKeyEscape) And &H80) <> 0 Then End
    Call wait(100)
    If Int(Rnd * 10000) < lv Then
      box(0) = Int(Rnd * 8)
      If numflag(box(0)) = False Then
        numflag(box(0)) = True
        time(box(0)) = 0
      End If
    End If
    box(4) = 0
    For f = 0 To 8
      If (GetKeyState(f + 97) And &H80) <> 0 And keyflag(f) = False Then
        keyflag(f) = True
        If numflag(f) = True Then
          score = score + 1
          lv = lv + 2
          Picture1(f).Cls
          numflag(f) = False
        Else
          miss = miss + 1
        End If
      ElseIf (GetKeyState(f + 97) And &H80) = 0 And keyflag(f) = True Then
        keyflag(f) = False
      End If
      If numflag(f) = True Then
        box(4) = box(4) + 1
        time(f) = time(f) + lineupspeed
        Picture1(f).Cls
        Picture1(f).Line (0, Picture1(f).ScaleHeight)-Step(Picture1(f).ScaleWidth, -
time(f)), &HFF, BF
        Picture1(f).CurrentX = 0: Picture1(f).CurrentY = 0
        Picture1(f).Print f + 1
        If time(f) > Picture1(f).ScaleHeight Then
          numflag(f) = False
          Picture1(f).Cls
          miss = miss + 1
          time(f) = 0
        End If
      End If
    Next f
    Me.Line (0, Label1.Top)-Step(Me.Width, -miss * (20 + (Me.Height - Label1.Height -
(Me.Height - Me.ScaleHeight)) / MAXMISS)), &HFF, BF
    If miss >= MAXMISS Then Call gameover: gameflag = False: miss = MAXMISS
    If box(4) = 0 And Mode(4).Checked = True Then Call Mix
    If miss <> 0 Or score <> 0 Then Label1.Caption = "Score:" & score & "/Miss:" & miss &
"/" & MAXMISS
    If score / LVUPAVERAGE >= gamelv Then
      gamelv = gamelv + 1
      lineupspeed = lineupspeed + UPLINESPEED
    End If
  Wend
  ModeSelect.Enabled = True
  Owari.Enabled = True
Loop

End Sub

Private Sub gameover()

  Dim missf As Integer

  Call wait(500)
```

```
    For missf = 0 To 8
      Picture1(missf).Cls
      numflag(missf) = False
      Picture1(missf).Left = Fix(missf Mod 3) * Picture1(missf).Width + Fix(missf Mod 3) * 50
      Picture1(missf).Top = Fix(missf / 3) * Picture1(missf).Height + Fix(missf / 3) * 50
    Next missf
    Picture1(0).Print "Gam"
    Picture1(1).Print "e O"
    Picture1(2).Print "ver"
    Picture1(3).Print "Sco"
    Picture1(4).Print "re:"
    Picture1(6).Print Mid(Str(score), 1, 2)
    Picture1(7).Print Mid(Str(score), 3, 2)
    Picture1(8).Print Mid(Str(score), 5, 2)
    Label1.Caption = "Push Enter to Restart"

End Sub

Private Sub Mix()

    Dim mixf As Long

    If Mode(0).Checked = True Then
      For mixf = 1 To 8
        Picture1(mixf).Left = Fix(mixf Mod 3) * Picture1(mixf).Width + Fix(mixf Mod 3) * 50
        Picture1(mixf).Top = Fix(mixf / 3) * Picture1(mixf).Height + Fix(mixf / 3) * 50
      Next mixf
      For mixf = 0 To 2
        box(3) = Picture1(mixf).Top
        Picture1(mixf).Top = Picture1(mixf + 6).Top
        Picture1(mixf + 6).Top = box(3)
      Next mixf
    ElseIf Mode(1).Checked = True Then
      For mixf = 1 To 8
        Picture1(mixf).Left = Fix(mixf Mod 3) * Picture1(mixf).Width + Fix(mixf Mod 3) * 50
        Picture1(mixf).Top = Fix(mixf / 3) * Picture1(mixf).Height + Fix(mixf / 3) * 50
      Next mixf
      For mixf = 0 To 2
        box(3) = Picture1(mixf).Top
        Picture1(mixf).Top = Picture1(mixf + 6).Top
        Picture1(mixf + 6).Top = box(3)
      Next mixf
      For mixf = 0 To 2
        box(2) = Picture1(mixf * 3).Left
        Picture1(mixf * 3).Left = Picture1(mixf * 3 + 2).Left
        Picture1(mixf * 3 + 2).Left = box(2)
      Next mixf
    ElseIf Mode(2).Checked = True Then
      For mixf = 1 To 8
        Picture1(mixf).Left = Fix(mixf Mod 3) * Picture1(mixf).Width + Fix(mixf Mod 3) * 50
        Picture1(mixf).Top = Fix(mixf / 3) * Picture1(mixf).Height + Fix(mixf / 3) * 50
      Next mixf
    ElseIf Mode(3).Checked = True Then
      For mixf = 1 To 8
        Picture1(mixf).Left = Fix(mixf Mod 3) * Picture1(mixf).Width + Fix(mixf Mod 3) * 50
        Picture1(mixf).Top = Fix(mixf / 3) * Picture1(mixf).Height + Fix(mixf / 3) * 50
      Next mixf
      For mixf = 0 To 2
        box(2) = Picture1(mixf * 3).Left
        Picture1(mixf * 3).Left = Picture1(mixf * 3 + 2).Left
        Picture1(mixf * 3 + 2).Left = box(2)
      Next mixf
    Else
      For mixf = 0 To MIXFREQUENCY
        box(0) = Int(Rnd * 8)
        box(1) = Int(Rnd * 8)
        box(2) = Picture1(box(0)).Left
        box(3) = Picture1(box(0)).Top
        Picture1(box(0)).Left = Picture1(box(1)).Left
        Picture1(box(0)).Top = Picture1(box(1)).Top
        Picture1(box(1)).Left = box(2)
        Picture1(box(1)).Top = box(3)
      Next mixf
    End If

End Sub

Private Sub wait(waittime As Long)

    tm = GetTickCount
    While tm + waittime > GetTickCount
      DoEvents
    Wend

End Sub

Private Sub Form_Resize()

    If Me.WindowState = 1 Or wide = 0 Then Exit Sub

    Me.Width = wide
    Me.Height = high
    Me.Left = (Screen.Width - Me.Width) / 2
    Me.Top = (Screen.Height - Me.Height) / 2

End Sub

Private Sub Mode_Click(Index As Integer)

    For f = 0 To 4
      Mode(f).Checked = False
    Next f
    Mode(Index).Checked = True

End Sub

Private Sub Form_Unload(Cancel As Integer)

    End

End Sub

Private Sub Owari_Click()

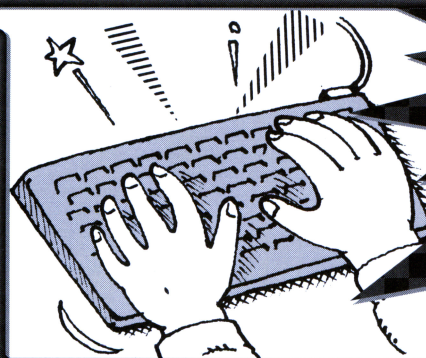
    End

End Sub
```

★はじめまして。HSP始めました。ポケコンも使っています！（大阪市・広岡聡之16歳）……【編：プログラム言語のフリーソフトの中で一番人気のHSPですが，けっこう，学校のパソコンにもインストールされているケースを見かけます。本誌連載もよろしく！】

入力方法は30ページをご覧ください。

Windows(Visual Basic V6.0)用



左右の文字をタイピング

L-R Typing（エル-アール タイピング）

松本　輝彦（13歳）

内容

画面に出てくる文字をキーボードで打っていく，いわゆるタイピング・ゲームです。右手で打つキーと，左手で打つキーが別々に出てきます(**写真２**)。

遊び方・操作方法

タイトル画面(**写真１**)でスペースボタンを押すと，ゲームが始まります。

上から文字が出てくるので，出てきた文字のキーを打ってください(**写真３**)。キーが合っていれば１点増え，間違えると１点減点されます。

制限時間は30秒です。制限時間になると得点が表示されます(**写真４**)。中央上に総得点，左右にそれぞれの得点が表示されます。真ん中の数字は，関係ないキーを打った数(ミス数)です。

入力方法

まず，ピクチャボックス(PicBack，PicMain)を二つ配置して，Scalemodeプロパティを「3 - ピクセル」に設定してください。その他のプロパティは，厳密な設定をしなくても大丈夫なのですが，適切な形で実行したい場合は，掲載されている**プロパティ表**どおりに設定してください。プロパティを設定したあとは，掲載されている**リスト**を入力してください。

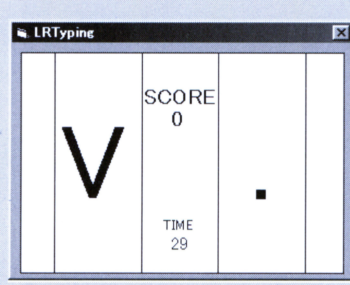
改造のアドバイスとしては，音を付ける，結果から弱点のキーを分析する…など，いろいろ出来ると思います。

最後に

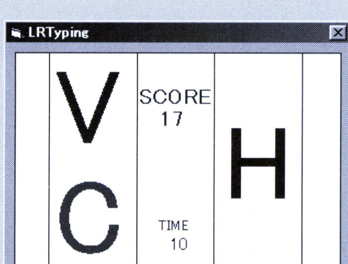
これで５回目の投稿になります。初めてタイピング・ゲームに挑戦しましたが，



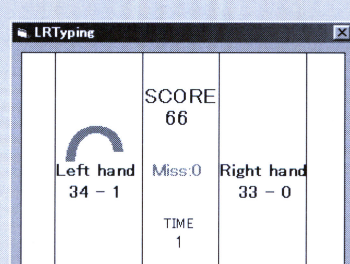
《写真１》左右の手で別々にタイピングを行うゲーム



《写真２》左と右に表示された文字を入力していく



《写真３》ルーレット風に次の文字が表示されてくる



《写真４》得点表示も左手，右手別々に表示してくれる

CHECKER FLAG

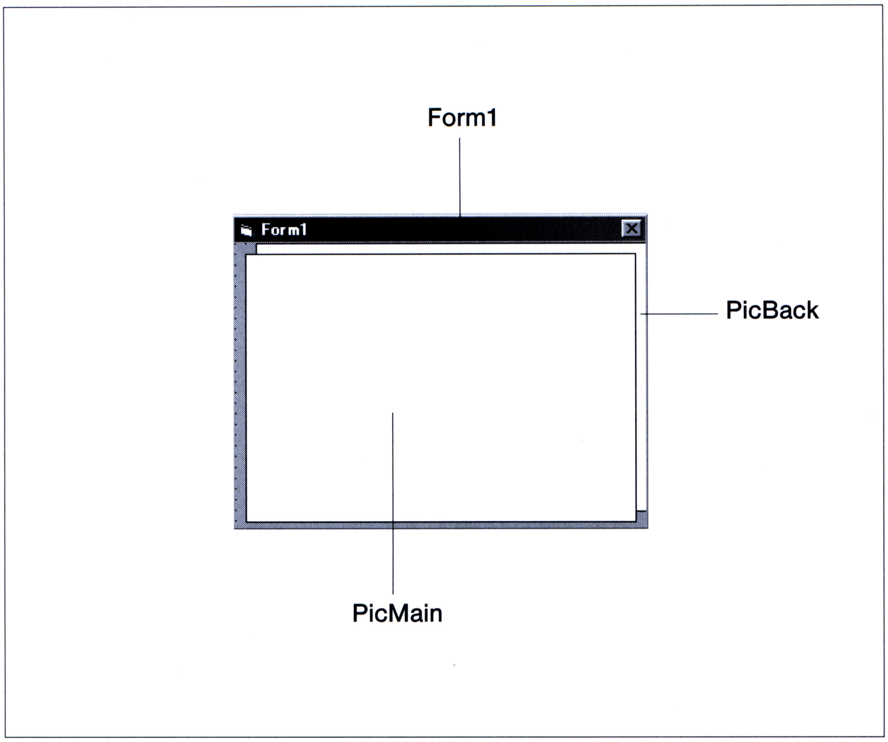
Dr.D：VB作品では，もはや常連の松本クンの作品じゃな。基本的なタイピングなので，全体的に単純な作りじゃが，左右の文字をタイピングするというアイデアが良い。リスト面でもキー入力部分で，この左右判定に合わせた記述の工夫が見られる。間違えたキーの左右の位置に合わせて文字の色が赤くなっている点に注目してもらいたい。

編：松本さんの作品の特徴は，単純なゲームでも何かひとつ個性をもっている点ですね。今回の作品も，ありきたりのタイピングゲームながら，左右の文字を別々にタイプしていくという独自のアイデアが個性を生み出しています。

つぐ美：しかもプログラム自体は，そんなに難しい処理を行っていないのよね。まさにベーマガ向けの作品作りのツボを得ているって感じね。

★今回も同じく「どうして次号発売日を誌面に表記してくれないんですか？」。僕はしてほしいです。目次のページがいいと思います。
（岡山市・大塚　均31歳）……【**編**：毎月8日発売ですが，わかりました，ご要望におこたえして，今月号から。目次をご覧ください。】

入力方法は30ページをご覧ください。



《第1図》フォームの配置図

友達からの評判が良かったので，送ってみました。評価よろしくお願いします。

メールアドレスは，mteruhiko@hotmail.com，それからホームページは，http://kumamoto.cool.ne.jp/teru0820 です。

《第1表》プロパティ表

フォームの設定	
Name	Form1
AutoRedraw	-1 'True
BorderStyle	1 '固定(実線)
Caption	Form1
Height	3570
Left	0
LinkTopic	Form1
MaxButton	0 'False
MinButton	0 'False
ScaleHeight	213
ScaleMode	3 'ﾋﾟｸｾﾙ
ScaleWidth	312
StartUpPosition	3 'Windows の既定値
Visible	0 'False
Top	0
Width	4770
ピクチャボックスの設定	
Name	PicMain
Appearance	0 'ﾌﾗｯﾄ
AutoRedraw	-1 'True
BackColor	&H80000005&
ForeColor	&H80000008&
Height	202
Left	8
ScaleHeight	200
ScaleMode	3 'ﾋﾟｸｾﾙ
ScaleWidth	295
TabIndex	1
Top	8
Width	297
Name	PicBack
Appearance	0 'ﾌﾗｯﾄ
AutoRedraw	-1 'True
BackColor	&H80000005&
ForeColor	&H80000008&
Height	202
Left	16
ScaleHeight	200
ScaleMode	3 'ﾋﾟｸｾﾙ
ScaleWidth	295
TabIndex	0
Top	0
Visible	0 'False
Width	297

《リスト》L-R Typingのプログラム・リスト

```
Option Explicit
Private Declare Function GetTickCount Lib "kernel32" () As Long
Private Declare Function BitBlt Lib "gdi32" (ByVal hDestDC As Long, ByVal x As Long, ByVal y As
Long, ByVal nWidth As Long, ByVal nHeight As Long, ByVal hSrcDC As Long, ByVal xSrc As Long,
ByVal ySrc As Long, ByVal dwRop As Long) As Long
Dim KeyLR(1) As String        '左右の文字振り分け
Dim MNum(1) As Byte           '現在表示されている文字ナンバー
Dim NxY(1) As Integer         '次の文字のY座標（落ちてくる
Dim ScY(1) As Integer         '押した文字のY座標（画面から消える
Dim ScMsg(1) As String        '押した文字
Dim KeyMsg(1) As String       '現在表示されている文字
Dim Score As Integer          '総得点
Dim LRSc(1) As Integer        '左右の得点
Dim Miss(2) As Integer        '左右，それ以外のミス数
Dim MTime(1) As Byte          'ミス時の赤い時間カウント
Dim TM As Long                'タイム
Dim KU As Byte                'キーが押されているかのフラグ
Dim GM As Byte                'ゲームの状態
Dim F As Byte                 'ループ用

'起動時
Private Sub Form_Load()
Randomize
Me.Show
'キーの振り分け
KeyLR(0) = "QWERTASDFGZXCVB"
KeyLR(1) = "YUIOPHJKLNM,."
GM = 0
Me.Caption = "LRTyping"
Call Msg("L-RTYPING", 30, 32, 36, 8)
Call Msg("L-RTYPING", 28, 30, 36, 0)
Call Main
End Sub

'メインルーチン
Sub Main()
TM = 30 * 20 '最初の30を変えると秒数が変わる
'文字描画
Call Msg("PRESS SPACE KEY", 41, 151, 18, 0)
Call Msg("PRESS SPACE KEY", 40, 150, 18, 15)
BitBlt PicMain.hDC, 0, 0, 290, 200, PicBack.hDC, 0, 0, &HCC0020
PicMain.Refresh
While GM = 0: DoEvents: Wend 'スペースが押されるまで待つ
Score = 0
'出てくる文字をランダムに決める
MNum(0) = Int(Rnd * 15) + 1
MNum(1) = Int(Rnd * 13) + 1
'初期設定
For F = 0 To 1
  KeyMsg(F) = Mid(KeyLR(F), MNum(F), 1)
  NxY(F) = -50
  Miss(F) = 0
  LRSc(F) = 0
Next F
Miss(2) = 0
KU = 1
'メインループ
Do
Call Wait(50)
Call Draw
TM = TM - 1
  '文字の移動
```

★ノーコメント強化月間。（岩手県釜石市・**鉄人坂井24歳**）……【**影**：……………。**つぐ美**：どうしたの影さん。**影**：ノーコメント！ つぐ美：今月は，影さんが何もしゃべらないそうよ。**編**：よし，ここからは，私とつぐ美ちゃんで…。**影**：ノーコメント時間終わりました。】

★コンピュータ・ウイルスの撃退法がとても参考になりました。（新潟県長岡市・南　寿和20歳）……【編：ウイルスの話題は昨年末から尽きませんね。Dr.D：うむ，一般の人たちの多くが，パソコンのセキュリティに関心を持ちはじめたのはうれしいことじゃないか。】

```
  For F = 0 To 1
  MTime(F) = (MTime(F) - (MTime(F) <> 0)) Mod 10
  ScY(F) = ScY(F) - (ScY(F) <> 205) * 25
  NxY(F) = NxY(F) - (NxY(F) <> 50) * 25
  Next F
Loop Until TM = 0 'タイムが0になったら抜ける
Beep
Call Draw
'太字で左右それぞれの点数，ミス数を描画
PicBack.Line (31, 100)-(109, 135), QBColor(15), BF
PicBack.Line (181, 100)-(259, 135), QBColor(15), BF
PicBack.Font.Bold = True
Call Msg("Left hand", 33 - F, 101 - F, 12, 0)
Call Msg("Right hand", 183 - F * 1, 101 - F, 12, 0)
Call Msg(Str(LRSc(0)) & " - " & Miss(0), 46 - Len(Str(LRSc(0))) * 2 - F, 121 - F, 12, 0)
Call Msg(Str(LRSc(1)) & " - " & Miss(1), 201 - Len(Str(LRSc(1))) * 2 - F, 121 - F, 12, 0)
Call Msg("Miss:" & Miss(2), 121 - F, 101 - F, 12, 12)
PicBack.Font.Bold = False
BitBlt PicMain.hDC, 0, 0, 290, 200, PicBack.hDC, 0, 0, &HCC0020
PicMain.Refresh
Call Wait(5000) '5秒待つ
GM = 0
Call Main
End Sub

'画面の描画
Sub Draw()
PicBack.Cls
PicBack.Line (30, -1)-(110, 200), , B
PicBack.Line (180, -1)-(260, 200), , B
'すでに押された文字（青）を描く
Call Msg(ScMsg(0), 35, ScY(0), 72, 9)
Call Msg(ScMsg(1), 190 + (ScMsg(1) = "I" Or ScMsg(1) = "," Or ScMsg(1) = ".") * -20, ScY(1),
72, 9)
'次に押す文字を描く
Call Msg(KeyMsg(0), 35, NxY(0), 72, (MTime(0) <> 0) * -12)
Call Msg(KeyMsg(1), 190 + (KeyMsg(1) = "I" Or KeyMsg(1) = "," Or KeyMsg(1) = ".") * -20,
NxY(1), 72, (MTime(1) <> 0) * -12)
'現在の総得点を描く
PicBack.Font.Bold = True
Call Msg("SCORE", 113, 30, 14, 0)
Call Msg(Str(Score), 143 - (Len(Str(Score))) * 6, 50, 14, 0)
PicBack.Font.Bold = False
'残りタイムを描く
Call Msg("TIME", 130, 150, 10, 0)
Call Msg(Str(TM ¥ 20 + 1), 142 - (Len(Str(TM ¥ 20))) * 3, 165, 12, 0)
PicBack.ForeColor = QBColor(0)
BitBlt PicMain.hDC, 0, 0, 290, 200, PicBack.hDC, 0, 0, &HCC0020
PicMain.Refresh
End Sub

'文字の描画
Sub Msg(MS As String, MX As Integer, MY As Integer, FS As Byte, FC As Byte)
With PicBack
  .CurrentX = MX
  .CurrentY = MY
  .ForeColor = QBColor(FC)
  .FontSize = FS
End With
PicBack.Print MS
End Sub

'ウェイト
Sub Wait(T As Integer)
Dim ST, AT
ST = GetTickCount
AT = ST
While AT - ST < T: DoEvents: AT = GetTickCount: Wend
End Sub

'フォーム閉じたら終了
Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As Integer)
End
End Sub

'キーが押されたときの処理
Private Sub PicMain_KeyDown(KeyCode As Integer, Shift As Integer)
Dim KeyTF As Byte      '押したキーの合否
'スペースが押されたとき
If KeyCode = 32 And GM = 0 And KU = 0 Then
GM = GM + 1
ElseIf GM = 1 And KU = 0 Then 'キーが離されていれば
 'キー判定
  For F = 0 To 1
   'キーが正しいとき
    If KeyCode = Asc(KeyMsg(F)) + (KeyMsg(F) = "," Or KeyMsg(F) = ".") * -144 Then
    KeyTF = F
    Exit For
    'キーが違うとき
    ElseIf KeyLR(F) Like "*" & Chr(KeyCode + (KeyCode = 188 Or KeyCode = 190) * 144) & "*"
Then
    KeyTF = F + 2
    Exit For
    '範囲外のキーを押したとき
    Else
    KeyTF = 4
    End If
  Next F
  '押したキーが合っていれば
  If KeyTF < 2 Then
  ScMsg(KeyTF) = Mid(KeyLR(KeyTF), MNum(KeyTF), 1)
  ScY(KeyTF) = 50
  MNum(KeyTF) = Int(Rnd * (15 - KeyTF * 2)) + 1
  KeyMsg(KeyTF) = Mid(KeyLR(KeyTF), MNum(KeyTF), 1)
  NxY(KeyTF) = -50
  Score = Score + 1
  LRSc(KeyTF) = LRSc(KeyTF) + 1
  'キーが間違っているとき
  ElseIf KeyTF = 2 Or KeyTF = 3 Then
  Score = Score + (Score <> 0)
  Miss(KeyTF - 2) = Miss(KeyTF - 2) + 1
  MTime(KeyTF - 2) = 1
  'その他のキーが押されたとき
  Else
  Score = Score + (Score <> 0)
  Miss(2) = Miss(2) + 1
  MTime(0) = 1
  MTime(1) = 1
  End If
End If
KU = 1
End Sub

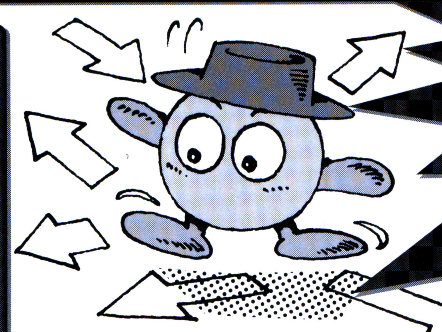
'キーが離されたときの処理
Private Sub PicMain_KeyUp(KeyCode As Integer, Shift As Integer)
KU = 0 'フラグを戻す
End Sub
```

入力方法は30ページをご覧ください。

Windows(Visual Basic V4.0)用

移植版

PC98の投稿作品をVBへ移植



BALL & ARROWS

（ボール アンド アローズ）

内田寛（52歳）

内容

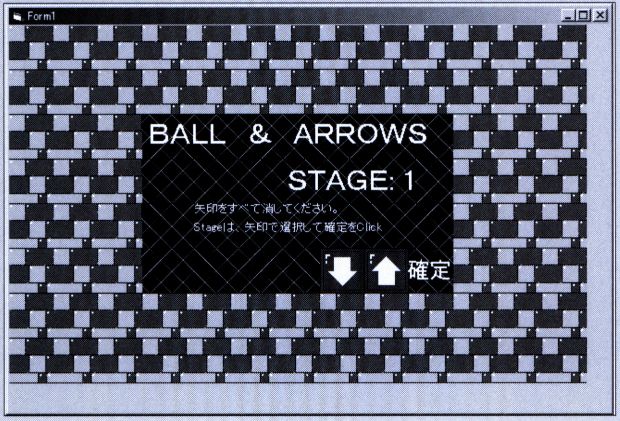
本誌2000年６月号に掲載された，PC98 N88-BASIC用『BALL & ARROWS』のVB移植版です。

操作方法

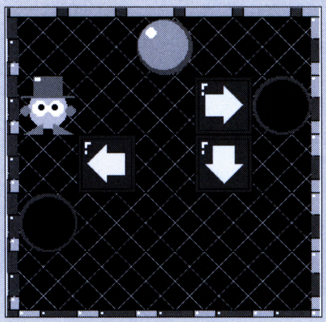
すべての動作をマウスのクリックで行います。

まず，タイトル画面に表示される矢印でステージを選択し，確定をクリックします(**写真１**)。ステージ画面に移った後，画面右側の矢印のどれかをクリックし，ボールの進行方向を決めると，ゲームがスタートします(**写真２**)。

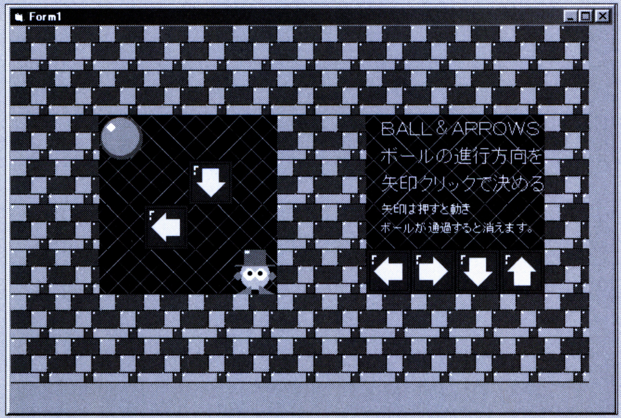
キャラクタは，上下左右いずれかのマ



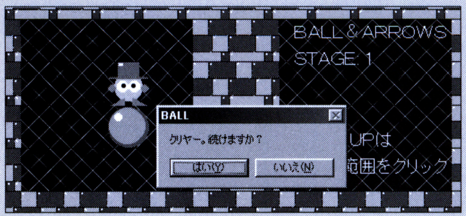
《写真１》PC98版の印象をそのまま引き継いでいる



《写真３》ステージが進むと穴などのトラップも出現



《写真２》ボールは壁に当たって跳ね返り続ける



《写真４》ステージ上の矢印をすべて消せばステージクリア

★業務連絡：何や恥ずかしくなったからPN変えました。／久々に愛読者カード送ってみました。どーでしょう？／もーすぐ春休み。久しぶりにプログラム組もっかな？（大阪府河内長野市・**よっち改めザ☆よしお19歳**）……【**影**：私もジョニーに…って，やめときます。】

スをクリックして動かしていきます。キャラクタは1マスずつ移動します。ステージにある矢印は，キャラクタが隣り合っている状態で矢印のマスをクリックすることで動かします。

ボールは，ブロックや人に当たると跳ね返ります。ステージによっては落とし穴があるので，ボールが穴に落ちないようにしていきます(**写真3**)。

ステージ上の矢印が，すべてなくなればステージクリアです(**写真4**)。

入力方法

掲載されている**プロパティ表**，**フォームの配置図**を参考にフォームを作成します。Picture1には画像ファイルを設定する必要があります。画像ファイルはフォームの配置図に出ている画像を参考に作成してください。1キャラクタの大きさが48×48になります。

フォーム作成後，掲載されている**リスト**を入力してください。

今回の移植の方法

ゲーム本編のプログラムの制作は，動きを理解したら原作にとらわれることなく，自分なりに処理方法を考えた方が効率的だと感じました。

制作手順は，はじめに方針を考えて，今回は「すべての動作をマウスのクリックで行う。Timerを使う。Case文などを使って少し長くなってもわかりやすく書く」と決めました。すべての動作をクリックでゲームができるように，フラグ(Fの値で示される)を使って三つの場面に分け，Mouse_Upの中で分岐しています。

ステージデータは6段を1列で書いたので，リスト側から少しでもステージ構成をわかりやすくするように9で区切りを入れました。ステージデータの内容は，次のようになります。0：床，1 2 3 4：矢印（左右下上），5：ブロック，6：穴，7：球，8：キャラクタ，9：区切り。

PC98との違いとして，17〜26面を新しく加えました。自分で付け加えるときは，付け加えたステージ数に合わせて，DIMのS$(26)とSTM＝26の値を変えてください(この場合，26がステージ数を示している)。

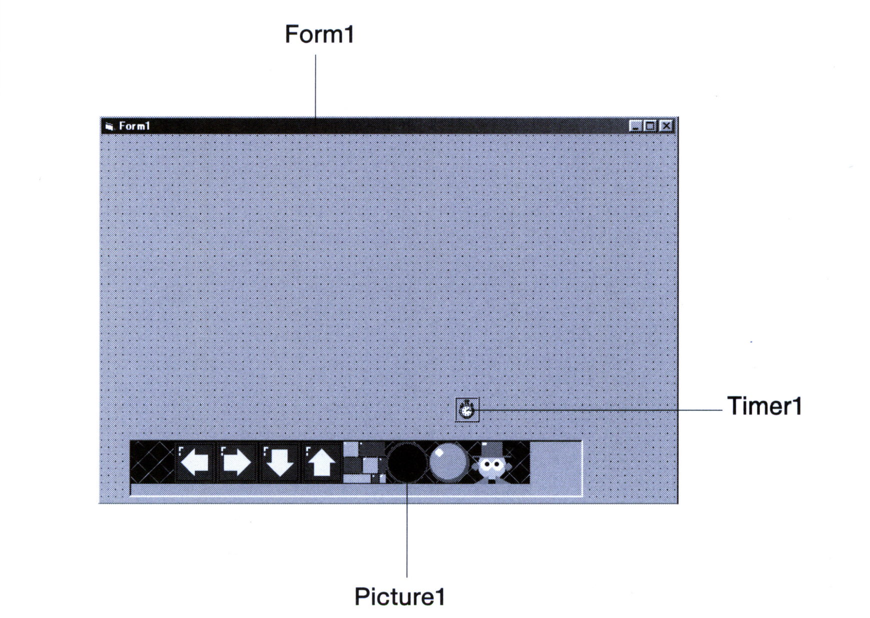
CHECKER FLAG

編：PC98の投稿作品の移植版です。以前にもコメントしましたが，命令が異なる言語間での移植は技術的に評価できますね。また，新しいステージが追加されているのも好感がもてます。

つぐ美：気になったのが，マウスクリックによる操作ね。キャラクタを動かすタイプのゲームは，マウスでなくキーボードのカーソルキーやテンキーで操作したほうが快適に感じますからね。

Dr.D:ゲームにおいて操作性というのは，ゲーム自体の印象に深く関わってくるため，しっかりと考慮してきてほしいものじゃな。ちと面倒な形式ではあるが，VBでのプログラムができる読者は，ぜひともキーボードへの改造に挑戦してほしいものじゃな。

影：ちなみに，タイトルでのステージセレクトを行う矢印の対応が逆になっていたため，その部分を修正しておいたゾ。



《第1図》フォームの配置図

《第1表》プロパティ表

フォームの設定	
Name	Form1
AutoRedraw	-1 'True
Caption	Form1
Height	6615
Left	1545
LinkTopic	Form1
PaletteMode	1 'Zｵｰﾀﾞｰ
ScaleHeight	414
ScaleMode	3 'ﾋﾟｸｾﾙ
ScaleWidth	649
Top	2265
Width	9855
ピクチャボックスの設定	
Name	Picture1
AutoRedraw	-1 'True
Height	65
Left	32
Picture	(ﾋﾞｯﾄﾏｯﾌﾟ)
ScaleHeight	'61
ScaleMode	3 'ﾋﾟｸｾﾙ
ScaleWidth	509
TabIndex	0
Top	344
Visible	0 'False
Width	513
タイマーの設定	
Name	Timer1
Interval	50
Left	400
Top	296

★はじめて買いました…さっぱりわかりません（笑）でしたが，「PIECE」は楽しそうですね。小型のMSXみたいな感じで，買ってみようかと思います。いつかBMに送りたいです。（宮崎県北諸県郡・JEY YAMA31歳）……【編：初心者にもわかりやすい記事を！ですね。】

《リスト》BALL & ARROWSのプログラム・リスト

```
Private Declare Function BitBlt Lib "gdi32" (ByVal hDestDC As Long, ByVal X As
Long, ByVal Y As Long, ByVal nWidth As Long, ByVal nHeight As Long, ByVal hSrcDC
As Long, ByVal xSrc As Long, ByVal ySrc As Long, ByVal dwRop As Long) As Long
Dim ST As Integer, F As Integer, YA As Integer, STM As Integer, OST As Integer
Dim BX As Integer, BY As Integer, KX As Integer, KY As Integer, TX As Integer
Dim TY As Integer, TXD As Integer, TYD As Integer, Map(12, 7), S$(26)

Private Sub Gamen1()
Dim I As Integer, J As Integer
For I = 0 To 7: For J = 0 To 12: Kaku J, I, 1, 5: Next J, I
For I = 2 To 5: For J = 3 To 9: Kaku J, I, 1, 0: Next J, I
Kaku 7, 5, 2, 3: Hyouji
End Sub

Private Sub Gamen2()
Dim I As Integer, J As Integer, M As Integer
For I = 1 To 6: Kaku 7, I, 1, 5: Next
For I = 2 To 5: For J = 8 To 11: Kaku J, I, 1, 0: Next J, I
 FontSize = 15
 CurrentX = 400: CurrentY = 100: Print "ＢＡＬＬ＆ＡＲＲＯＷＳ"
 CurrentX = 400: CurrentY = 130: Print "ボールの進行方向を"
 CurrentX = 400: CurrentY = 160: Print "矢印クリックで決める"
 CurrentX = 400: CurrentY = 190: FontSize = 10: Print "矢印は押すと動き"
 CurrentX = 400: CurrentY = 210: Print "ボールが通過すると消えます。"
YA = 0: For K = 1 To 41
I = K Mod 7: J = Int(K / 7) + 1
 M = Val(Mid$(S$(ST), K, 1)): Map(I, J) = M
  Select Case M
      Case 1, 2, 3, 4 '矢印、YAで数を数える
        Kaku I, J, 1, M: YA = YA + 1
      Case 0 '床
        Kaku I, J, 1, 0
      Case 5 'ブロック
        Kaku I, J, 1, 5
      Case 6 '穴
        Kaku I, J, 1, 6
      Case 7 '球     (TX、TY) 球の座標
        Kaku I, J, 1, 7: TX = I: TY = J
      Case 8 '人     (BX、BY) 人の座標
        Kaku I, J, 1, 8: BX = I: BY = J
     End Select
Next
 Kaku 8, 5, 4, 1 '（８，５）から４つの矢印をかく
End Sub


Private Sub Hyouji()
 ForeColor = QBColor(15): FontSize = 23
 CurrentX = 150: CurrentY = 100: Print "ＢＡＬＬ　＆　ＡＲＲＯＷＳ"
 CurrentX = 300: CurrentY = 150: ForeColor = QBColor(0): Print "ＳＴＡＧＥ:";
OST
 CurrentX = 300: CurrentY = 150: ForeColor = QBColor(15): Print "ＳＴＡＧＥ:";
ST
 CurrentX = 430: CurrentY = 250: FontSize = 18: Print "確定"
 CurrentX = 200: CurrentY = 190: FontSize = 10: Print "矢印をすべて消してくださ
い。"
 CurrentX = 200: CurrentY = 210: Print "Stageは、矢印で選択して確定をClick"
End Sub


Private Sub Kaku(A As Integer, B As Integer, C As Integer, D As Integer)
  BitBlt Form1.hDC, A * 48, B * 48, C * 48, 48, Picture1.hDC, D * 48, 0,
&HCC0020
  Refresh
End Sub

Private Sub Form_Load()
Timer1.Enabled = False: ST = 1: STM = 26 '最大面数
 S$(1) = "5555559570005950030595010059500085955555"
 S$(2) = "5800009500026950473095610009500000955555"
 S$(3) = "0056009003000970001595200089000400900650"
 S$(4) = "0050009020010950803595107059040020900050"
 S$(5) = "0000039052050900801090207009050350940000"
 S$(6) = "5560009008050903031090120409050700900065"
 S$(7) = "8000009015010900504090405009020530900000"
 S$(8) = "5506559570015930300394002019530085955605"
 S$(9) = "0600009080130903500090705409042000900006"
S$(10) = "0000059025400900810090047009001530950000"
S$(11) = "6017039005540900020098010009035500940000"
S$(12) = "0100609430000900070293080009000014906003"
S$(13) = "8000069010300930203090274049004040960000"
S$(14) = "6000089003210920030093470049000400921010"
S$(15) = "0068309300000900210090032009000004904760"
S$(16) = "2000069103030900084090270009040104960000"
S$(17) = "6080209202000900075493030009010026940000"
S$(18) = "0020009040051900580090424549300270900600"
S$(19) = "0400209005050905004090400059057252902000"
S$(20) = "0010094235049600010900870090160109006040"
S$(21) = "6024409003313906570090080009440000902000"
S$(22) = "6001069000003932403191071009230480960104"
S$(23) = "6400069010342900072090320169405000910200"
S$(24) = "0020069300101910070290064239028000960401"
S$(25) = "6040209025010950800090103009700300941006"
S$(26) = "0010090530009000010903268490000719200406"
F = 1: Gamen1 'ST は STAGE数、９は区切り
For I = 0 To 7: For J = 0 To 7: Map(I, J) = 5: Next J, I
End Sub
```

★コンピュータ・エンターテインメント・サイドがなくなったのは残念である。現在のソフト事情を知ることができなくなったからである。（福岡県行橋市・**河野哲哉28歳**）……【**影**：読者ニーズや時代の変化，編集側の先を読んだ意図など，いろいろあるわけですな。】

Visual Basic

入力方法は30ページをご覧ください。

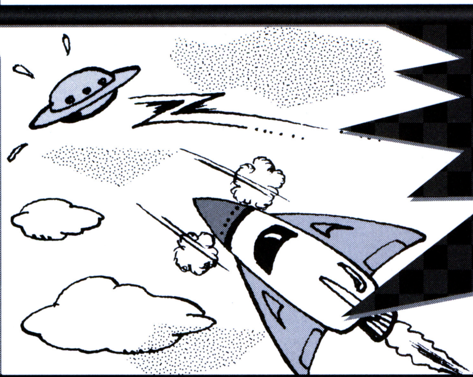
★「投稿プログラムのメール」を間違えて同じ内容のもの2回送ってしまいました。たいへん失礼いたしました……。それから，メールに部分に何を書いたらいいものか，いつも迷います。白紙でいいの？それも失礼だし。（東京都練馬区・**畑の木29歳**）……**【影：うーむ。】**

```
Private Sub Form_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As
Single)
Dim I As Integer, J As Integer, C As Integer, D As Integer
KX = Int(X / 48): KY = Int(Y / 48) 'クリックした座標
Select Case F
Case 1 'Ｆ＝１はタイトル画面
 If KX = 8 Then
   ST = ST + 1: OST = ST - 1: If ST = STM + 1 Then ST = STM
   Hyouji
 ElseIf KX = 7 Then
   ST = ST - 1: OST = ST + 1: If ST = 0 Then ST = 1
   Hyouji
 ElseIf KX = 9 Then 'Stage確定
   F = 2: Gamen2
 End If
Case 2 'Ｆ＝２は球の方向を決める面（ゲームスタート画面）
 If KX = 8 Then TXD = -1: TYD = 0 '左方向
 If KX = 9 Then TXD = 1: TYD = 0 '右方向
 If KX = 10 Then TXD = 0: TYD = 1
 If KX = 11 Then TXD = 0: TYD = -1
 If KX >= 8 And KX <= 11 Then
  F = 3: Timer1.Enabled = True
  For I = 2 To 5: For J = 8 To 11: Kaku J, I, 1, 0: Next J, I
 End If
 FontSize = 15
 CurrentX = 400: CurrentY = 100: Print "ＢＡＬＬ＆ＡＲＲＯＷＳ"
 CurrentX = 400: CurrentY = 130: Print "ＳＴＡＧＥ:"; ST
 CurrentX = 400: CurrentY = 220: Print "ＧＩＶＥ　ＵＰは"
 CurrentX = 400: CurrentY = 250: Print "ここの範囲をクリック"
Case 3 'Ｆ＝３はゲーム中
 If KX >= 8 Then    'Give Up
   If MsgBox("もう一度チャレンジ？", vbYesNo) = vbYes Then Timer1.Enabled =
False: F = 2: Gamen2: Exit Sub Else End
 End If
 If Abs(BX - KX) <= 1 And Abs(BY - KY) <= 1 Then
   C = Map(KX, KY): D = Map(2 * KX - BX, 2 * KY - BY)
   If C = 0 And Abs(BX - KX) + Abs(BY - KY) = 1 Then
     Map(BX, BY) = 0: Map(KX, KY) = 8  '人の移動
     Kaku BX, BY, 1, 0: Kaku KX, KY, 1, 8
     BX = KX: BY = KY
   ElseIf C >= 1 And C <= 4 And D = 0 And Abs(BX - KX) + Abs(BY - KY) = 1 Then
     Map(BX, BY) = 0: Map(KX, KY) = 8  '矢印の移動
     Map(2 * KX - BX, 2 * KY - BY) = C
        Kaku BX, BY, 1, 0: Kaku KX, KY, 1, 8
        Kaku 2 * KX - BX, 2 * KY - BY, 1, C: BX = KX: BY = KY
   End If
 End If
```

```
End Select
End Sub


Private Sub Timer1_Timer()
FF = 0: B = Map(TX + TXD, TY + TYD)
Select Case B '進む先は？
 Case 0, 1, 2, 3, 4 'Ｂ＝０床
    Kaku TX, TY, 1, 0: Map(TX, TY) = 0
    TX = TX + TXD: TY = TY + TYD
    Kaku TX, TY, 1, 7: Map(TX, TY) = 7
    Select Case B
      Case 1 'Ｂ＝１は左矢印
        TXD = -1: TYD = 0: YA = YA - 1
      Case 2 'Ｂ＝２は右矢印
        TXD = 1: TYD = 0: YA = YA - 1
      Case 3 'Ｂ＝１は下矢印
        TXD = 0: TYD = 1: YA = YA - 1
       Case 4 'Ｂ＝１は上矢印
        TXD = 0: TYD = -1: YA = YA - 1
     End Select
 Case 5, 8, 9 'Ｂ＝５はブロック、８は人
      TXD = -TXD: TYD = -TYD
 Case 6 'Ｂ＝６は穴
      FF = 1: Kaku TX, TY, 1, 0
End Select
If YA = 0 Then   '矢印がゼロ
  Timer1.Enabled = False
  If ST = STM Then
    If MsgBox("最後のSTAGEをクリアーしました。初期画面に戻る？", vbYesNo) = vbYes
Then F = 1: Gamen1: Exit Sub Else End
  Else
    If MsgBox("クリヤー。続けますか？", vbYesNo) = vbYes Then ST = ST + 1: F = 2:
Gamen2: Exit Sub Else End
  End If
End If
If FF = 1 And YA <> 0 Then 'FF＝１は穴に落ちた
  Timer1.Enabled = False
  If MsgBox("穴に落ちました。もう一度チャレンジ？", vbYesNo) = vbYes Then F = 2:
Gamen2: Exit Sub Else End
End If
End Sub
```

Windows(Visual C++)用



破片による連鎖破壊に注目

Dust（ダスト）

kou（19歳）

Visual C++

入力方法は32ページをご覧ください。

はじめに

最近は，DirectXを使ったゲームがインターネット上に増えたような気がします。なので，今回はその流れに逆らい(?)，「C++とWindowsSDKのみで，どれくらいのシューティングゲームが作れるのか？」にチャレンジしてみました。

ソースには，かなりコメントを入れてしまったので，ちょっと見にくくなったかもしれません。

内容

見た目は普通のシューティングゲーム。しかし，敵を破壊すると敵の破片が飛び散り，その破片が敵に当たると，また破片が飛び散るという連鎖型シューティングゲームです。

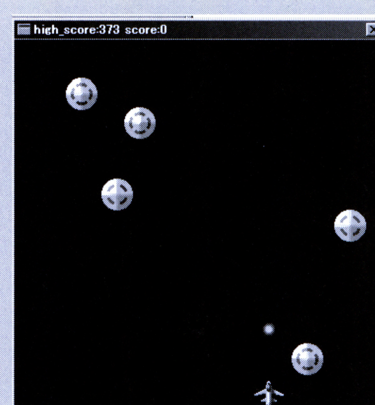
遊び方

タイトル画面でエンターキーを押すとゲームスタート(**写真1,2**)。

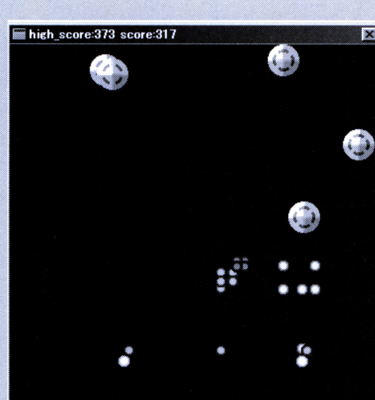
ゲーム中はカーソルキーの左右で移動。スペースキーで弾が撃てます。弾の連射



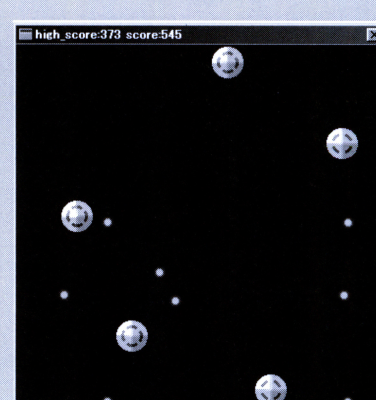
《写真1》敵の破片処理が特徴のシューティング



《写真2》画面内を動き回る敵を倒すと…



《写真3》破片が飛び散る。この破片にも当たり判定がある



《写真4》自機がやられても破片があるうちは，得点が加算される

★95，98のOS探しています。誰か譲ってください。説明書があるとたいへん助かります。(静岡県清水市・**瑠璃香27歳**)……【**影**：さて，どうしたものか。おっ！いい方法があった！探す苦労をWindowsXPを入手する手段に向けることだ。我ながらgood！】

はできません。

自機がやられても，連鎖が続いている限りゲームオーバーにはなりません(**写真3,4**)。

プログラム入力方法

VisualC++ならば，空のWin32Applicationのプロジェクトを作成し，dust.cpp(**リスト1**)，dust.h(**リスト2**)，dust.rc(**リスト3**)ファイルを作成後，対応しているリストを打ち込んでください。

dust.rcファイルがエディタで作成できない時は，メモ帳などのテキストエディタで入力して作成してください。

それと，メニューのプロジェクト－設定－リンクのオブジェクトの項目にwinmm.libを書き加える事を忘れないでください。

改造のアドバイス

あまりにもタイトルや背景などが手抜きなので，星を上から下に流したり，最大連鎖数なんかが出るようにすれば，もうちょっと面白くなると思います。

あとは，ソースの中にも書いてますが，Cnum，Dnumの値などを変えて，難易度調整をすると良いと思います。

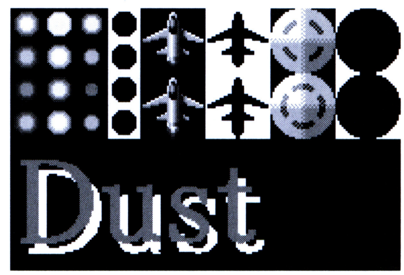
苦労した点

コード打ちの時は，あまり苦労はありません。ただ，MFCやDirectXの資料は沢山あるに,Windows SDKの資料がちょっと少なかったので，いろいろ調べるのに手間どりました。特にバックサーフェイスの作り方など。

終わりに

C++&WindowsSDKで作って，「結構使えるな」という感じがしました。家のパソコン(Pentium Ⅱ133MHz，Windows98)でも，サクサクと動きました。

今回，わかりやすくコメントをたくさん入れようと思っていたら，いつの間にかリスト容量が10KBを超えてしまいました。しかし，行数では中くらいの長さだと思うので，超大作で送っていません。いくらくらいが，超大作なのでしょうか？



dust-bmpの画像

CHECKER FLAG

Dr.D：この作品はぜひともリストに注目してほしい。処理の関数化，構造体の適切な使用などC言語プログラムとしては，非常にキレイな形でリストが記述されておるのじゃ。キャラクタの移動処理も変数内に保存してある移動量を加算するという簡単な処理ながらも，その移動量の設定が工夫されておるため，キレイな動きを実現しておるのじゃ。また，コメントの記述がしっかりしている点も，他の投稿者に見習ってほしいところじゃな。

編：リスト面での解説を補足すると，このゲームの特徴である破片の当たり判定は，再帰処理でなくBurst関数内でのwhileによるループで行っていますね。

つぐ美：説明文にある超大作の目安についてですが，リスト容量では15KBくらいが目安と言っていますが，これは，行数で言えば500行くらいになりますかね。

《リスト1》dust.cppのリスト

```
#include "dust.h"

char key[256]; //キー入力用
char text[256]; //文字列用
//改造は下の４つの数字をいじると良いかも。
#define Cnum 25 //敵キャラの数
#define Dnum 120 //弾（１個で固定）＆破片の数(Cnum＊５くらいが適量)
#define SizeX 380 //ウィンドウの横幅
#define SizeY 380 //ウィンドウの縦幅
//スコア
long score = 0, high_score = 0;
struct Chara {//自機と敵の構造体
        double x;//X座標
        double y;//Y座標
        double xstep;//X座標の移動幅
        double ystep;//Y座標の移動幅
        int pict;//アニメーション
        int Flag;//データの状態（０なら待機、１なら自機、２なら敵）
};
//０は自機、１～Cnumまでが敵のデータ。
Chara chara[Cnum + 1];

struct Dust {//弾＆破片の構造体
        double x;//X座標
        double y;//Y座標
        double xstep;//X座標の移動幅
        double ystep;//Y座標の移動幅
        int pict;//アニメーション
        double value;//
        int Flag;//データの状態（０なら待機、１は赤、２は黄、３は青）
};
//０は自機の弾、１～Dnumまでが破片のデータ
Dust dust[Dnum + 1];

HBRUSH hBrush, hBrushOld;//画面塗りつぶし用のブラシのハンドル
```


★昨年，専門学校の非常勤講師を頼まれルンルン気分で行ったところ，いきなり「VB」をやれと言われ，今までFortran，BasicをMS-DOSでしかやったことのない私は大慌て！しかも15年以上もコンピュータ離れ。(北海道札幌市・**伊藤元和62歳**)……**【編：続報を。】**

```
HDC backdc, memdc;//デバイスコンテキストのハンドル
HBITMAP hBitmap, backBitmap;//ビットマップハンドル
HINSTANCE hInst;
RECT rect;

//関数の宣言
int Appear(int);
void keyin(void);
void Move(void);
void Burst(int, int, int, double);
void Draw(HWND, HDC, int);
void Syokika(void);

//==========バックサーフェイス(backdc)の作成========
void CreateBackDC(HWND hwnd, HDC hdc)
{
	backdc = CreateCompatibleDC(hdc);
	memdc = CreateCompatibleDC(NULL);

	GetClientRect(hwnd, &rect);
	backBitmap = CreateCompatibleBitmap(hdc, SizeX, SizeY);
	SelectObject(backdc, backBitmap);
	//バックバッファの初期化(黒く塗りつぶす)
	hBrush = CreateSolidBrush(RGB(0, 0, 0));
	hBrushOld = (HBRUSH)SelectObject(backdc, hBrush);
	PatBlt(backdc, 0, 0, SizeX, SizeY, PATCOPY);
	//ビットマップの取得
	hInst = (HINSTANCE)GetWindowLong(hwnd, GWL_HINSTANCE);
	hBitmap = LoadBitmap(hInst, "DUST");
	SelectObject(memdc, hBitmap);
}

//===============画面描画===============
void Draw(HWND hwnd, HDC hdc, int state)
{
	int i;
	//バックサーフェイスの塗りつぶし
	PatBlt(backdc, 0, 0, SizeX, SizeY, PATCOPY);
	//自機&敵の描画
	for (i = 0; i <= Cnum; i++) {
		if (chara[i].Flag != 0) {
			BitBlt(backdc, (int)chara[i].x - 16, (int)chara[i].y - 16,
				32, 32, memdc, chara[i].Flag*64 + 32, chara[i].pict * 32,
SRCAND);
			BitBlt(backdc, (int)chara[i].x - 16, (int)chara[i].y - 16,
				32, 32, memdc, chara[i].Flag*64, chara[i].pict * 32,
SRCINVERT);
		}
	}
	//弾&破片の描画
	for (i = 0; i <= Dnum; i++) {
		if (dust[i].Flag != 0) {
			BitBlt(backdc, (int)dust[i].x - 8, (int)dust[i].y - 8,
				16, 16, memdc, 48, 0, SRCAND);
			BitBlt(backdc, (int)dust[i].x - 8, (int)dust[i].y - 8, 16, 16,
				memdc, (dust[i].Flag - 1)*16, dust[i].pict * 16,
SRCINVERT);
		}
	}
	if (state == 0)
		BitBlt(backdc, (SizeX - 128) / 2, SizeY / 2 - 64, 128, 64,
			memdc, 0, 64, SRCCOPY);

	sprintf(text, "high_score:%d score:%d", high_score, score);
	SetWindowText(hwnd, text);
	//バックサーフェイスからメイン画面に転送
	BitBlt(hdc, 0, 0, SizeX, SizeY, backdc, 0, 0, SRCCOPY);
}

//ウィンドウプロシージャ
HRESULT CALLBACK WndProc(HWND hwnd, UINT msg, WPARAM wP, LPARAM lP)
{
	switch (msg) {
		case WM_DESTROY://終了時の後片付け
			DeleteDC(memdc);
			DeleteDC(backdc);
			DeleteObject(hBitmap);
			DeleteObject(hBrush);
			PostQuitMessage(0);
			break;
		case WM_KEYDOWN://キーが押された
			key[wP] = 1;
			break;
		case WM_KEYUP://キーが離された
			key[wP] = 0;
			break;
		default:
			return DefWindowProc(hwnd, msg, wP, lP);
	}
	return 0;
}

//メイン関数
int WINAPI WinMain(HINSTANCE hThisInst, HINSTANCE hPrevInst,
						LPSTR lpCmdLine, int nShowCmd)
{
	int i = 0, j = 0;
	int game_state = 0;
	//敵の出現用タイム
	int a_time = 0;

	HWND hwnd;
	MSG msg;
	WNDCLASS wc;
	HDC hdc;
	//ウィンドウのクラス
	char szWinName[] = "dust";
	wc.lpszClassName = szWinName;
	wc.hInstance = hThisInst;
	wc.hCursor = LoadCursor(NULL, IDC_ARROW);
	wc.hIcon = LoadIcon(NULL, IDI_APPLICATION);
	wc.lpfnWndProc = WndProc;
	wc.style = 0;
	wc.lpszMenuName = "IDR_MEUN";
	wc.cbClsExtra = 0;
	wc.cbWndExtra = 0;
	wc.hbrBackground = (HBRUSH) GetStockObject(WHITE_BRUSH);
```

★久々に投稿します。つい最近PIECEを買ったので、これに関する記事を書いてください。お願いします。(高知市・**銀猫25歳**)……【**編**：PIECEネタ多くありませんか？**影**：ペーマガでも1月号と2月号で紹介しましたが、その効果かな？**編**：また短期連載考えますか。】

Visual C++
入力方法は32ページをご覧ください。

★お久しぶりです。コンピュータ・ウイルスの話はたいへん参考になりました。私も気をつけたいと思います。ホントに。（兵庫県加古川市・きのえくま25歳）……【編：1，2月号連続「ウイルス」企画のウケがいいです。つぐ美：私は風邪のウイルスにまいってます。】

```
	if (!RegisterClass(&wc)) return 0;
	//ウィンドウの作成
	hwnd = CreateWindow(szWinName, "dust", WS_OVERLAPPED | WS_SYSMENU,
		CW_USEDEFAULT, CW_USEDEFAULT, SizeX, SizeY + 20, HWND_DESKTOP,
		NULL, hThisInst, NULL);
	//ウィンドウの表示
	ShowWindow(hwnd, nShowCmd);
	UpdateWindow(hwnd);

	hdc = GetDC(hwnd);//ウィンドウのデバイスコンテキストハンドルの取得
	CreateBackDC(hwnd, hdc);//バックサーフェイスの作成

	DWORD tim;
	srand(time(NULL));//擬似乱数ジェネレータを初期化する。

	Syokika();

	while (1) {//メインループ
		tim = timeGetTime();//時間の取得

		if (game_state == 0) {//タイトル
			if (key[13])//エンターキーでスタート
				game_state = 1;
			score = 0;
		}
		else if (game_state == 1) {//ゲーム中
			a_time = Appear(a_time);//敵出現
			keyin();//ゲーム中のキー処理
			Move();//移動処理＆当たり判定
			if (chara[0].Flag == 0) {
				for (i = 0; i <= Dnum; i++)
					if (dust[i].Flag == 0) j++;
				if (j == Dnum + 1) {
					if (high_score < score) high_score = score;
					game_state = 0;
					Syokika();
				}
				j = 0;
			}
		}
		Draw(hwnd, hdc, game_state);//描画
		do{//メッセージループ
			while(PeekMessage(&msg, NULL, 0, 0, PM_NOREMOVE)){
				if(!GetMessage(&msg, NULL, 0, 0))
					exit(0);
				TranslateMessage(&msg);
				DispatchMessage(&msg);
			}
		}while(timeGetTime() < tim + 20);
		//0.020秒経ったらメッセージループを抜ける
	}
	return msg.wParam;
}
//===============敵出現================
int Appear(int time)
{
	time++;
```

```
	if (time >= 25) {
		for (int i = 1; i <= Cnum; i++) {
			if (chara[i].Flag == 0) {
				chara[i].x = rand() % (SizeX - 36) + 16;
				chara[i].y = 16;
				chara[i].xstep = rand() % 3 + 1;
				chara[i].ystep = rand() % 3 + 1;
				chara[i].pict = 0;
				chara[i].Flag = 2;
				time = 0;
				break;
			}
		}
	}
	return time;
}
//===============キー処理================
void keyin()
{	//移動幅
	if (key[37]) chara[0].xstep = -4;
	else if (key[39]) chara[0].xstep = 4;
	else chara[0].xstep = 0;
	//弾を発射する
	if (chara[0].Flag != 0 && key[32]) {
		if (dust[0].Flag == 0) {
			dust[0].Flag = 1;
			dust[0].x = chara[0].x;
			dust[0].y = chara[0].y - 16;
			dust[0].xstep = 0;
			dust[0].ystep = -10;
			dust[0].pict = 0;
			dust[0].value = 1;
		}
	}
}
//==============移動＆当たり判定=============
void Move()
{
	int i, j;
	//キャラクターの動き
	for (i = 0; i <= Cnum; i++) {
		if (chara[i].Flag != 0) {
			if (16 <= chara[i].x + chara[i].xstep
				&& chara[i].x + chara[i].xstep <= SizeX - 20)
				chara[i].x += chara[i].xstep;
			else
				chara[i].xstep = - chara[i].xstep;

			if (16 <= chara[i].y + chara[i].ystep
				&& chara[i].y + chara[i].ystep <= SizeY - 16)
				chara[i].y += chara[i].ystep;
			else
				chara[i].ystep = - chara[i].ystep;
			if (i != 0)//敵キャラは下向きに加速
				chara[i].ystep += 0.1;

			chara[i].pict++;//アニメーション
			if (chara[i].pict == 2)
```

```
                                        chara[i].pict = 0;
                              //敵との当たり判定
                              if (chara[0].Flag != 0 && i != 0)
                                        if (chara[0].x-16 <= chara[i].x && chara[i].x <=
chara[0].x+16
                                        && chara[0].y-16 <= chara[i].y && chara[i].y <=
chara[0].y+16)
                                        Burst(0, (int)chara[0].x, (int)chara[0].y,
dust[i].value);
                    }
          }
          //弾＆破片の動き
          for (i = 0; i <= Dnum; i++) {
                    if (dust[i].Flag != 0) {
                              dust[i].x += dust[i].xstep;
                              dust[i].y += dust[i].ystep;

                              if (i != 0)//破片は下に向けて加速
                                        dust[i].ystep += 0.1;

                              dust[i].pict++;//アニメーション
                              if (dust[i].pict == 4)
                                        dust[i].pict = 0;
                              //当たり判定
                              for (j = 0; j <= Cnum; j++) {
                                if (chara[j].Flag != 0)
                                  if (chara[j].x-12 <= dust[i].x && dust[i].x <=
chara[j].x+12
                                        && chara[j].y-12 <= dust[i].y && dust[i].y <=
chara[j].y+12)
                                            Burst(j, (int)chara[j].x, (int)chara[j].y,
dust[i].value);
                              }
                              //画面の上か下に消えるとデータを待機状態にする。
                              if (dust[i].y <= - 10 || dust[i].y >= SizeY)
                                        dust[i].Flag = 0;
                    }
          }
}
//==============炸裂の処理===============
void Burst(int num, int x, int y, double val)
{
          int i = 4;//爆発時の飛び散る破片の数。
          int j = 1;//破片データは１番から（０番は弾）

          score += long(val * 10);
          chara[num].Flag = 0;//当たったキャラクターを待機状態に
          //破片を４つ出すか、全ての破片データを調べ終えるまでループ
          while(i >= 1 && j <= Dnum) {
                    if (dust[j].Flag == 0) {//待機状態のデータがあれば
                              dust[j].x = x;
                              dust[j].y = y;
                              dust[j].pict = 0;

                              if (num == 0) dust[j].Flag = 2;//自機の爆発なら黄色
                              else dust[j].Flag = 3;//敵なら青色の破片

                              if (i >= 3) dust[j].xstep = 2;
                              else dust[j].xstep = - 2;
                              if (i % 2 == 0) dust[j].ystep = -3;
                              else dust[j].ystep = 0;

                              dust[j].value = 1.2 * val;
                              i--;
                    }
                    j++;
          }
}

void Syokika()
{
          int i;
          //自機の初期設定
          chara[0].x= SizeX / 2;
          chara[0].y = SizeY - 24;
          chara[0].ystep = 0;
          chara[0].ystep = 0;
          chara[0].pict = 0;
          chara[0].Flag = 1;
          for (i = 1; i <= Cnum; i++)
                    chara[i].Flag = 0;
          //全ての弾＆破片を待機状態に
          for (i = 0; i <= Dnum; i++)
                    dust[i].Flag = 0;
}
```

《リスト２》dust.hのリスト

```
#include <windows.h>
#include <time.h>
#include <mmsystem.h>
#include <stdio.h>
#include <string.h>
#include <io.h>
#include <stdlib.h>

#define DUST 100
```

《リスト３》dust.rcのリスト

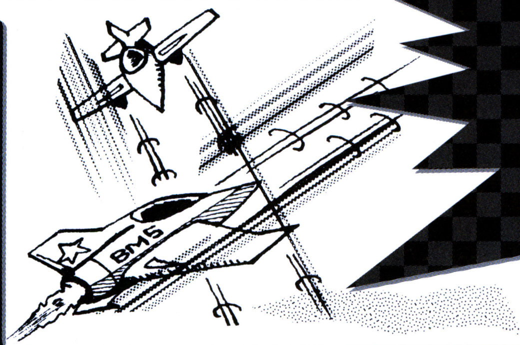
```
///////////////////////
//      BitMap       //
///////////////////////
DUST                BITMAP DISCARDABLE  "dust.bmp"
```

★今年に入ってから一度も「CMM試聴室」が掲載されていません。つぶれちゃったのですか？（埼玉県北葛飾郡・**鹿谷浩一19歳**）……**【つぐ美**：当面，休載ということになりました。また盛り上げたいですね。**編**：多くの人が参加できるもの，何かアイデアありませんか？**】**

入力方法は32ページをご覧ください。

リストが誌面に掲載できない作品です。
BMホームページ(http://www.basicmagazine.net)からダウンロードしてお楽しみください。

Windows(Visual C++)用



3D形式で処理されたシューティング

3D Shoot! Ver1.0

(スリーディー シュート)

片桐 貴志(23歳)

内容

3Dシューティングゲームです。3Dのゲームができたので，嬉しくなって本誌に初投稿しました。実行するには，DirectX7以降がインストールされている必要があります。ご注意ください。

描画はDirct3Dを使っていますが，はっきり言って2次元シューティングです。ただ，Aキーと Zキーを入力することによって視点変更ができます(これがミソ。っていうか,3Dなんだから当たり前か…)。

使い方・遊び方

タイトル画面でリターンキーを押すとゲーム開始です。

自機をカーソルキーで操ってください(写真1)。カーソルキーの↑で加速，↓で減速，←で左回旋，→で右回旋です。Aキーと Zキーで視点変更できます(写真2)。

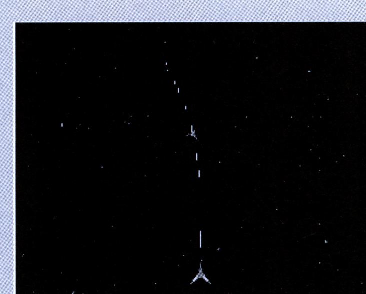
スペースキーで弾を発射して敵を撃って倒してください(写真3)。敵機に衝突するか，敵機の発射する弾に当たるかするとゲームオーバーです。



《写真1》見た目はそれっぽくないが、実は3D処理されている



《写真2》視点変更した画面。写真1と比べてみて


《写真3》飛び回っている敵機を打ち落とせ

★「ねぇ，人の脳ってROMだと思う？」「ROM…いやRAMに近いROMだね？」「フフ…そうねぇ…」(愛知県名古屋市・ズオウリアキ29歳)……【影：だから，これ，誰と誰が会話してるものなのかなあ？編：誰でもいいじゃない，ネタなんだから。影：寿司が食べたい！】

右下にはレーダーマップが表示されており，白い点が自機のいる位置で，赤い点が敵の座標です。

最初自機が光りますが，その間は無敵です。いきなり敵に当たって死ぬことが無いよう配慮しました。

敵を全部倒すとステージクリアです。でもステージクリアしても今のところ特に変わり映えするわけではなく，最初の敵の数が増えるだけです。ひたすら敵機を撃ち倒してください。

プログラムの仕組み

Direct3Dで固められているプログラムです。リストのほとんどがDirect3D関連のために使われています。そのためゲームの核となっている部分は，ほんの一部分だったりします。

ひとつ，このプログラムで凝ったところといえば，境界処理（x,z平面でxとzが境界マイナスとプラスのマックス値に近づくところ）で，敵キャラや玉の表示がおかしくならないようにした部分くらいですかね。

改造のアドバイス・苦労した点

リストにある最初のdefine値をいろいろいじってみてください。RANGEの値は少なくとも5000は欲しいところです(RANGEはフィールド全体の幅です。それより少なくしても構いませんが表示がおかしくなります。ただ，プログラムがどういう処理をしているのか見るのにはいいかも)。

速度が速いと感じる人は，星の数を増やしてください。逆に遅いと感じる人は星の数を減らしてください(最初のdefineのMAX_STAR値を加減してください)。

MAXV,MINVは自機の速度の上限と下限です。いろいろ変えてみてください。

苦労した点は，とにかくバグとの格闘に尽きます。機体が表示されたり表示されなかったり，玉がおかしく表示されたり…など。3Dなのでデバッグの苦労も3次元(謎)。あとはDirect3D 自体の理解に,かなり時間を費やされました。

CHECKER FLAG

編：一見すると3Dっぽくないんですが，視点変更があるところに3Dらしさを感じます。

つぐ美：ゲームとしては，もう少し爽快感があると良かった感じね。自機の移動の小回りが利かないので，なかなか敵に弾を当てられないのよね。

Dr.D：注目してほしい点は，右下のマップ表示じゃな。すべての敵機と自機の位置情報を，マップの大きさに合わせて常に表示しなければならないので，処理を実現させるにはチョットした工夫が必要なのじゃ。ただ残念なのが，パソコンの速度によっては，このマップ部分の描画にちらつきが生じてしまうことじゃな。

Visual C++
入力方法は32ページをご覧ください。

ゲリラコラム VBって凄いや 文／松原拓也

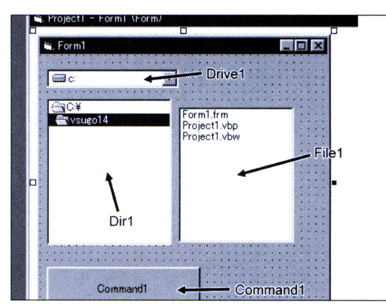
このコラムでは，Visual Basicのちょっとしたワンポイント・テクニックを紹介しています。

今月のポイント html自動生成

WEBサイトでファイルをダウンロードするとき，親切に「***k bytes」と表示されているページがよくありますね。今回は，そういうダウンロード用のページ（htmlファイル）を自動生成するプログラムを作ってみました。(約30分ほど経過)……はい，できました（写真1，リスト1）。使いかたは簡単で，作成したいフォルダ（写真2）を選んでボタンを押すだけです。すると，そのディレクトリに“temp.html”というファイルができあがります。ファイルにはファイルへのリンク，ファイルサイズが書き出されます（写真3）。

ファイルサイズを取得するには「FileLen」という関数を使います。外に，「Get」と「Seek」を使う方法（リスト2）がありますが，1バイトずつ読み込むため処理が遅くオススメしません。

このhtmlファイルを自動生成するというアイデアは，いろいろなことに応用できるのではないでしょうか？



▲《写真1》フォームの作成

《リスト1》html自動生成プログラム

```
Private Sub Command1_Click()
    fpath = Dir1.Path  ' フルパス名の取得
    If Right$(fpath, 1) <> "¥" Then
        fpath = fpath + "¥"
    End If

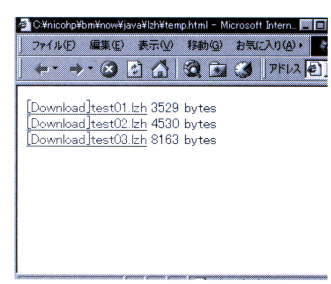
    Open fpath + "temp.html" For Output As #2 'ファイル作成
    Print #2, "<html><body>"
    For i = 0 To File1.ListCount - 1
        File1.ListIndex = i
    '    Iret = fncFileSize(fpath + File1.filename)
        Iret = FileLen(fpath + File1.filename) ' サイズの取得
        Print #2, "<a href=" + File1.filename + ">"
        Print #2, "[Download]" + File1.filename + "</a>"
        Print #2, Str$(Iret) + " bytes<br>"
    Next
    Print #2, "</body></html>"
    Close #2
End Sub

Private Sub Dir1_Change()
    File1.Path = Dir1.Path
End Sub

Private Sub Drive1_Change()
    Dir1.Path = Drive1.Drive
End Sub
```



▲《写真2》こんなファイルを元に……



▲《写真3》……このようなhtmlファイルが作られました

《リスト2》ファイルサイズの取得関数（処理が遅いです）

```
Function fncFileSize(fname As String) As Long
    Dim tmpbyte As Byte

    Open fname For Binary As #1 'バイナリでオープン
    While (EOF(1) = False)
        Get 1, , tmpbyte  '1バイトずつ読込み
    Wend
    Iret = Seek(1) - 2 '最終ポインタを取得
    Close #1
    fncFileSize = Iret
End Function
```

★読者ひとり増やしました！コーエーが命綱か！がんばれBM！がんばれ，つかち〜。（福井県武生市・つかち〜17歳）……【**編**：3月号のプレゼントのこと言ってるんですかね。**影**：そうだろうね。まあ，私が命綱と言ってる人もいるしな。**つぐ美**：いませんって。】

Pick Up LIST

《リスト１》マップ描画処理と視点変更処理のリスト

```
jiku=D3DXVECTOR3(0, 1, 0);
if(chr[0].exist==1)
        m_pMatrixStack->RotateAxis(&jiku, -chr[0].rot);
else
        m_pMatrixStack->RotateAxis(&jiku, 0);

jiku=D3DXVECTOR3(1, 0, 0);
//※視点変更処理を行う命令になる
m_pMatrixStack->RotateAxis(&jiku, rolex);
if(game==0)
{
        //右下の座標値出力
        //※レーダーマップ描画部分になる
        pPrimSurf=m_pD3DX->GetPrimary();
        pPrimSurf->GetDC(&hDC);
        fnWhiteBox(hDC, 540, 380, 640, 480);
        fnBlackBox(hDC, 541, 381, 639, 479);
        for(i=0;i<num+1;i++)
        if(chr[i].exist==1)
                //この命令で各機体の位置を点で表示する
                SetPixel(hDC,chr[i].dx*100/RANGE+50+540,-chr[i].dz*100/RANGE+50+380,
RGB(255,255*(i==0),255*(i==0)));

        pPrimSurf->ReleaseDC(hDC);

        //キー入力処理
        if(GetAsyncKeyState(VK_LEFT)!=NULL)
                chr[0].rot=chr[0].rot-0.02;
        if(GetAsyncKeyState(VK_RIGHT)!=NULL)
                chr[0].rot=chr[0].rot+0.02;
        if(GetAsyncKeyState('Z')!=NULL){
                rolex=rolex+0.01;
                if(rolex>-3.1415/2+3.1415/2/2)
                        rolex=-3.1415/2+3.1415/2/2;
        }
        if(GetAsyncKeyState('A')!=NULL){
                rolex=rolex-0.01;
                if(rolex<-3.1415/2-3.1415/2/2)
                        rolex=-3.1415/2-3.1415/2/2;
        }
```

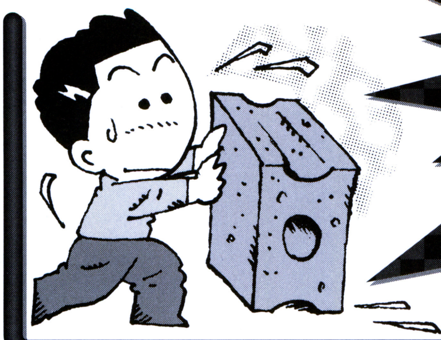
Dr.Dのワンポイント解説

Dr.D：CHECKER FLAGで触れているマップ描画部分で工夫がされておる点が，SetPixel命令の引数部分じゃ。マップの大きさに合わせた座標変換の計算を引数の部分で行うことで，機体の位置表示は，このSetPixel命令だけで行えるというわけじゃ。また，このゲームの目玉でもある視点変更は，カメラの移動ではなく，行列スタックを利用した形で行っておる。詳しい説明は省略するが，RotateAxis命令で指定した軸を中心に回転する行列を求めることで，視点変更処理をしておるのじゃ。

★3月号70ページで言ってた「オリジナル言語」，3月号掲載時点ですでに完成。…でも処理速度遅めでした。（青森県南津軽郡・ていこく22歳）……【影：何でも最初から完璧ということはないから，がんばって，納得のいくものに仕上げてください。ファイト！】

(超大作プログラム) リストが誌面に掲載できない作品です。
BMホームページ(http://www.basicmagazine.net)からダウンロードしてお楽しみください。

Windows(Delphi4 Standard)用



ブロックの移動方法が決め手

ぶろっくランド

らー抹茶（22歳）

入力方法は33ページをご覧ください。

ストーリー

ある陽気な昼下がり，ルンルン気分で散歩中のポムム君。ドジな事に崖から転落!! 気がつけば不思議な空間にいるではありませんか。

「ここは，どこだよぉ～～～～!!」と，言いつつも，彼はひたすらブロックを動かすのでした。

遊び方

オーソドックスなアクション・パズルゲームです。全部で17ステージあります。

ゲームを起動するとタイトルが表示されるので，リターンキーでゲームスタートです(**写真１**)。

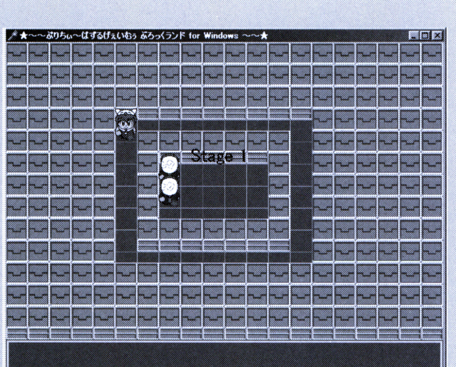
ポムム君をカーソルキーで操作，スペースキーでブロックを押します(**写真２**)。

ゲーム中，ESC キーでポーズ(一時停止)が出来ます。

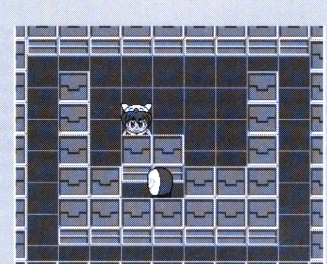
ステージ内を徘徊する謎の敵の倒し方ですが，ブロックとブロックの間に敵が挟まれた状態で，どちらかのブロックを敵のいる方向に押してやると，移動したブロックと移動方向にあるブロックに挟まれる状態で，敵を倒す事が出来ます(**写真３**)。ブロックに挟まれてない状態では敵は倒せません。

ブロックには色が２種類あり，青いブロックは，押すと１マスずつしか進みませんが，赤ブロックは，前方にブロックが無い限りどこまでも移動します。その際，移動方向に敵がいた場合，敵を倒していきます。その時の得点は２倍になります。

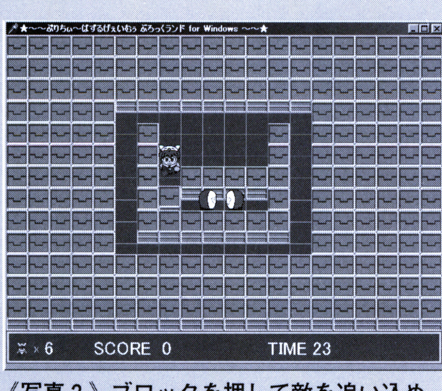
ゲームクリア条件は，ステージ内の敵をすべて倒すか，ゲーム中盤から出てく



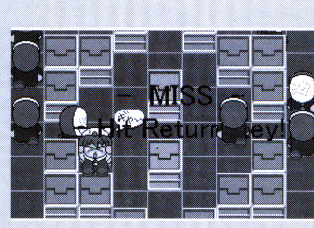
《写真１》スタンダードな作りのゲームと言える



《写真３》ブロックの間に挟んで敵を倒す



《写真２》ブロックを押して敵を追い込め



《写真４》敵に触れると１ミス。残機がなくなるとゲームオーバー

CHECKER FLAG

つぐ美：画像作成ソフトなど，掲載を意識した工夫があったのですが，本編のリストの容量から超大作扱いとさせていただきました。ただ，せっかくですので画像作成を行うソフトもダウンロードファイルに収録してあります。使用する必要はないのですが，プログラムの参考にしてくださいね。

影：ゲームをプレイしていて，NAMCOの名作『ロンパーズ』を思い出してしまった。

Dr.D：この手のタイプのゲームで出てくるのが配列を用いた仮想画面処理。VBの連載講座のアクションゲーム編で頻繁に解説したので，本誌をしっかり読んでいる読者にはもうお馴染みの処理じゃろう。

★去年の文月に受付完了したヤ○ーのADSLが未だに調整中。特別遅い地域みたいですが，それにしても…ねぇ。（東京都中央区・まぢけん20歳）……【編：たしかに，不平・不満は多いですね。やりたいときにできないというのが，つらいですよね。気力もなえちゃうし。】

る３つの鍵を揃えた場合にクリアとなります。

ゲーム後半から落とし穴や崩れる床などのトラップも出てきます。

また，ポムム君は敵に触れたり，穴に落ちた場合に１ミスとなります。画面左下に表示される自キャラの残り数がなくなると，ゲームオーバーです。

ステージクリア後には，経過時間ボーナスと取得キー数ボーナスが得点として入ります(**写真４**)。次のステージに進む場合は，リターンキーを押してください。

苦労した部分

キャラクタ同士の重ね合わせ処理に，一番苦労しました。移動時に敵やプレイヤーが，他のキャラクタの 頭の上を歩いたり，移動方向によっては重ね合わせがうまくいかなかったりと，いろいろな面で悩みました。そのためかプログラムは，あまり綺麗な物ではありません。何度か見直して修正したりしましたが，これくらいで勘弁してください…。

あとは，ステージ作成で苦労しました。

★参考文献

・「ここまでできるDelphi2.0パワーガイド」　小出俊夫 著 秀和システム

・「Delphi Object Pascalリファレンスブック」　(株)アンク著 技術評論社

・「Delphi Win32API リファレンス 」John Ayres 他 著，光田 秀 訳アスキー出版局

・「改訂版　Win32API オフィシャルリファレンス」マイクロソフト監修　アスキー出版局

Pick Up LIST

《リスト１》ブロック判定と描画処理部分のリスト

```
    {■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■}
{■  ﾌﾞﾛｯｸの移動配列処理                              }
{■引数    無し                                         }
{■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■}
Procedure TForm1.Mov_BlockHi();
Var cnt_x,cnt_y,cnt,M_fg:byte;
begin
  With PlyD[Block] do begin
    sx:=0;sy:=0;M_fg:=0;
    case anm_n of {ﾌﾞﾛｯｸの位置取得}
      1:begin sy:= 1;mov_t:=4;py:=py+32;end;
      2:begin sy:=-1;mov_t:=3;py:=py-32;end;
      5:begin sx:=-1;mov_t:=1;px:=px-32;end;
      6:begin sx:= 1;mov_t:=2;px:=px+32;end;
    end;
    cnt_y:=py div 32;cnt_x:=px div 32;c_dat:=Map_dat[cnt_y][cnt_x];

    {ﾌﾞﾛｯｸが有るかの判定}
    if (((cnt_y>0)and(cnt_y<12))and((cnt_x>0)and(cnt_x<19))
      and((Map_Dat[cnt_y][cnt_x]>7)and(Map_Dat[cnt_y+sy][cnt_x+sx]=0))) then begin

       if (Map_c[cnt_y+sy][cnt_x+sx]<>0)and(c_dat=RED_B) then {特殊ﾌﾞﾛｯｸ衝突判定}
         Enemy_Dead(2,Map_c[cnt_y+sy][cnt_x+sx]-1) {敵の死亡処理}
       {敵との衝突判定}
       else if (Map_c[cnt_y+sy][cnt_x+sx]<>0)and(Map_Dat[cnt_y+(sy*2)][cnt_x+(sx*2)]>4) then
          Enemy_Dead(1,Map_c[cnt_y+sy][cnt_x+sx]-1)
       else if Map_c[cnt_y+sy][cnt_x+sx]<>0 then M_fg:=1;{ﾌﾞﾛｯｸ移動時に敵がいると移動不可(挟まれていない)}
        if M_fg = 0 then begin
          m_ok:=False;
          Map_Dat[cnt_y+sy][cnt_x+sx]:=200; {ﾌﾞﾛｯｸの移動中ﾌﾗｸﾞ}
          Map_Dat[cnt_y][cnt_x]:=0;
          ox:=px div 32;oy:=py div 32;
          for cnt:=0 to 1 do begin {ﾏｯﾌﾟ障害物の描画}
            ImageList1.Draw(Make_bmp.Canvas,0,cnt*32-16,0);{床の描画}
            if (Map_dat[cnt_y+cnt-1][cnt_x]>4)and(Map_dat[cnt_y+cnt-1][cnt_x]<200)then
              ImageList1.Draw(Make_bmp.Canvas,0,cnt*32-32,Map_dat[cnt_y+cnt-1][cnt_x])
            else if(Map_dat[cnt_y+cnt-1][cnt_x]>0)and(Map_dat[cnt_y+cnt-1][cnt_x]<200)then
              ImageList1.Draw(Make_bmp.Canvas,0,cnt*32-16,Map_dat[cnt_y+cnt-1][cnt_x]);
          end;
        end
        else c_dat:=0;
      end
      else c_dat:=0;
  end;
end;
```

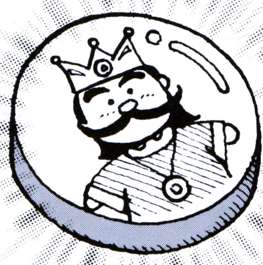
Dr.Dのワンポイント解説

Dr.D：リストのコメントがしっかりしているのでわかると思うが，この部分にあるif文でマップ上にブロック判定を行っている。if文内で仮想画面を表す２次元配列Map_Dat[][]の値の比較を行っているのがわかるはずじゃ。また，画面の描画も，このMap_Dat[][]の値から描画するブロックを判定しておるのじゃ。

★ヒゲを剃ったら「ターンA」。チゲなら「ソウル」でしょ。／「トトロ」の隣に「グリンチ」。影さんみたいだなあ。／店頭PC2台もフリーズ。お前はもう，ユリアッ～！／関西弁ON！我，難波しよっと！（愛知県名古屋市・**最馬場可？歳**）……【**影：いつになく壊れてます。**】

Windows(F-BASIC V6.3)用

定番ゲームをWindowsでプレイ



チェッカー2

ロディマス（29歳）

F-BASIC

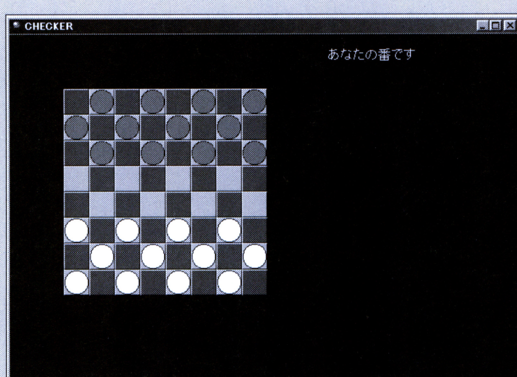
入力方法は31ページをご覧ください。

内容

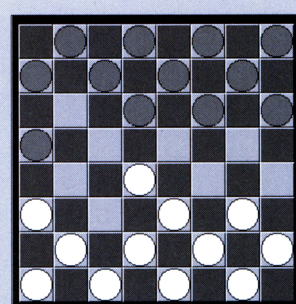
以前，PC98のN88-BASICで作った『チェッカー』(本誌2000年3月号掲載)をF-BASICへ移植し，さらにグラフィックとコンピュータ側の思考ルーチンを大幅に改良したものです。

プレイ方法

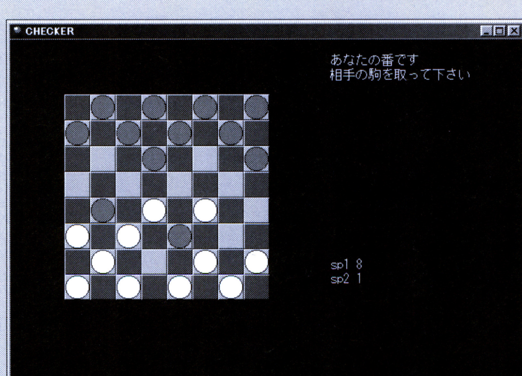
マウスを使用してゲームをプレイしま



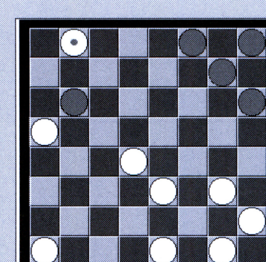
《写真１》西洋では定番のテーブルゲーム



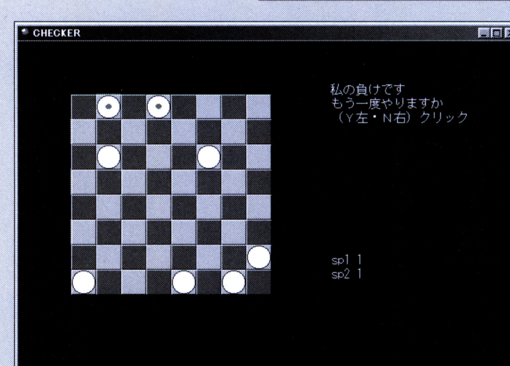
《写真２》駒を斜めに移動させて進めていく



《写真３》相手の駒をまたぐことで捕獲ができる



《写真４》相手の最前列へ移動するとキングに昇格



《写真５》相手の駒をすべて獲れば勝ちとなる

★僕のHPにはベーマガのバナーが貼ってあります。／受験やらゲームやらでなかなか正式公開に踏み切れないマイHP。／何事も基礎が肝心。BASICマガジン…いい響きです。／2キロほど肥えました。つぐ美さんは？（東京都練馬区・ルストフレム15歳）……【編：OK！】

す。着手は交互に行い，プレイヤーが先手になります(**写真1**)。自分の番になったら動かしたい駒を左クリックした後，移動先のマス目を左クリックしてください(**写真2**)。

駒についての説明

駒の動かし方

駒は，斜め前方に1マスずつ動かすことができます。キングに昇格するまで斜め後方への移動はできません。駒がすでにあるマス目には移動できません。

駒の獲り方

相手の駒をジャンプすると，その駒をキャプチャーする(獲る)ことができます(**写真3**)。ジャンプできるのは，斜め前方に相手の駒があって，その向こう側のマス目が空いているときだけです。キャプチャーした相手の駒は盤上からとりのけます。獲った駒の再使用はできません。

また，状況しだいでは数回のジャンプを行って，複数個の駒を獲ることができます。このような場合は連続ジャンプとなり，強制的に駒を移動させます。つまり，ジャンプできる駒があるのにジャンプしないことは許されません。ただし，ジャンプできる個所が二つ以上あるときは，好きなほうを選択します。

キングへの昇格

駒が前進して，相手の最前列に到達するとキングに昇格し(これを駒のプロモートと言う)，斜め後方にも動かせるようになります(**写真4**)。ただし，キングに昇格したときは，昇格することで,その回のムーブは終了します。

プレイの終了

負けが決まるのは次のケースです。
① 駒が全部獲られたとき(**写真5**)
② 動かせる駒がなくなったとき

入力方法

掲載されている**リスト**をF-BASICのエディタから入力し，実行します。ただし，プログラムの実行には画像ファイルが必要になります。掲載されている画像ファイル図を元にペイントなどで画像ファイルを作成してください。

必要な**画像ファイル**は6種類で，普通の状態とキングになった状態の駒の画像を，赤と白それぞれ用意します(計4種類になる)。あと，盤を描画するための画像を2種類用意します。この画像ファイルを作成する際には，WALL_.bmpの方のマスの色合いを濃くしてください。

前のプログラムとの違い

前のプログラムでは，キャプチャーを適当にやっていたのですが，このプログラムでは，駒を獲れるすべての手を調べ上げ，一番多くの駒を獲れる手を打つようになっています。このアルゴリズムを作るのは無茶苦茶むずかしかったけど，自慢できるところでもあります。

ただ，この処理を行っているわりには，適当に思考を行っている前のプログラムの方が強いみたいです…。今度は先読みを行い，もっと強いプログラムを作ってみせます。

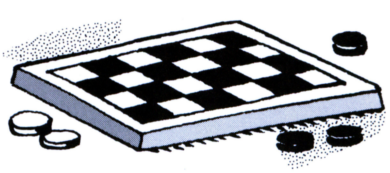
CHECKER FLAG

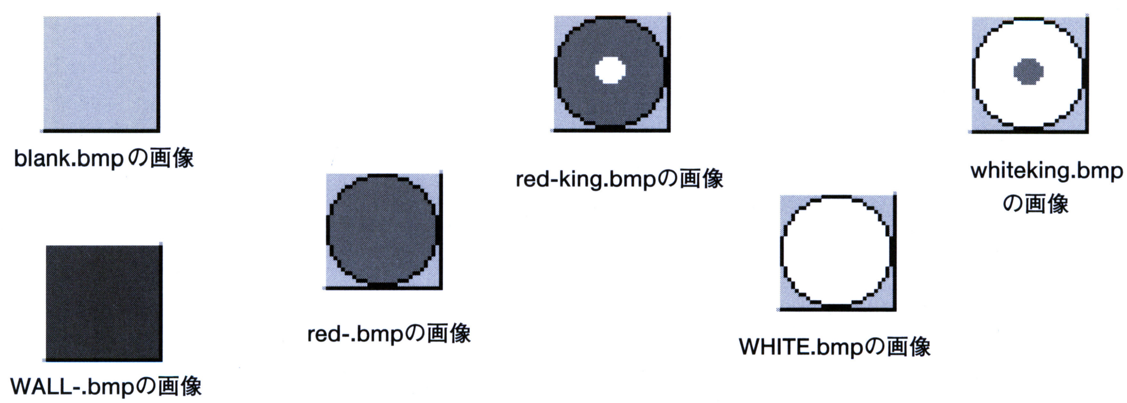
影：日本ではあまり馴染みがないかもしれないけど，西洋ではけっこう有名なテーブルゲーム。チェッカーを知らないという人は，ぜひともこのゲームから，チェッカーを知ってほしいものだ。

Dr.D:ロディマス君の作品を掲載するたびに言っている気もするが，やはりCOM処理を実現させるのがうまいのぉ。ただ，説明でも触れているように今回のCOMはあまり強くはない。じゃが，手を決めるまでのアルゴリズムはしっかりしているので，興味のある読者はこのCOMの強化改造なんかをやってみると面白いじゃろう。

編：細かい点ですが，PC98版で指摘したキャンセル動作が改善されていますね。

★参考にしたもの
・「世界のゲーム事典」松田道弘著　東京堂出版





《画像ファイル》

★僕も影さんみたいに影の薄い男になりたいです。(福岡市・**前田宣弘28歳**)……【**影**：こらこら，そんなに大きな声だとDr.に聞こえるでしょ。シーッ！**Dr.D**：何か？**影**：はっ！**Dr.D**：何やら「薄い」とか聞こえたが！？**影**：Dr.の頭の話じゃないですってば，ううぅ。】

《リスト》チェッカー2の プログラム・リスト

```
'##########################################################################
'# チェッカー
'# 皇紀２６６１年１０月より製作開始
'# 自慢は連続捕獲の処理をちゃんとやったこと
'#  （一番多く駒を取れる手を選びます）
'# 使用言語 F-BASICVer6.3
'# OS WINDOWS98-SECOND-EDITION
'##########################################################################
#include "windows.bi"
declare sub DISPLAY()                                    '盤と駒を表示
declare function COMPUTER(COL%)                         'コンピュータの思考ルーチン
declare function PLAYER(COL%)                           'プレイヤーの手番
declare function PROMOTION_CHECK(COL%,OLD%,NEW%)'成りのチェック
declare function COM_MOVE(COL%,OLD%,EV2%)               'キツネの動きを調べる
declare function COM_CAPTURE(COL%,OLD%,EV2%)            '捕獲を調べる
declare function COM_NEW_CHECK(COL%,I%,OLD%,LST%)       '移動先のマスを調べる
declare function MAN_INPUT_CHECK(IN_OR_OUT%)            'プレイヤーの入力
declare function MOUSEWAIT()                            'マウスの入力
declare function COM_CLIST_CHECK(COL%,I%,OLD%,LST%)     'コンピュータの移動をチェック
declare function MAN_JUMP_CHECK(COL%,OLD%,NEW%)         'プレイヤーの動きをチェック
declare function MAN_MOVE_CHECK(COL%,OLD%,NEW%)         'プレイヤーの動きをチェック
var shared BOARD(16*16+2) as integer            '盤
var shared COURSE(4,5) as integer               '進む方向
var shared BLANK as object                      '
var shared KING__RED as object                  '
var shared KINGWHITE as object                  '
var shared RED__ as object                      '
var shared WHITE as object                      '
var shared WALL_ as object                      '
var shared WIN_LOSE_CHECK as integer            '
var shared K_KING__RED as integer               '
var shared K_KINGWHITE as integer               '
var shared K___RED as integer                   '
var shared K_WHITE as integer                   '
var shared K__WALL as integer                   '
var shared K_BLANK as integer                   '
var shared SP as integer
'捕獲するアルゴリズムに使う
var shared LIST(20),NNEW(20),OOLD(20),AMNG(20),PRMT(20),ENMY(20) as integer
'乱数で打ち手を変える
var shared R_LST(20,20),R_NEW(20,20),R_OLD(20,20),R_AMG(20,20),R_PRM(20,20),R_ENM(20,20)
as integer
var shared MAX_CAPTURE_SP(20) as integer
BITMAPOBJECT WALL_
BITMAPOBJECT BLANK
BITMAPOBJECT RED__
BITMAPOBJECT WHITE
BITMAPOBJECT KING__RED
BITMAPOBJECT KINGWHITE
WALL_.LOADFILE "WALL_.BMP"
BLANK.LOADFILE "BLANK.BMP"
RED__.LOADFILE "RED__.BMP"
WHITE.LOADFILE "WHITE.BMP"
KING__RED.LOADFILE "RED__KING.BMP"
KINGWHITE.LOADFILE "WHITEKING.BMP"
COURSE(0,0)=-15:COURSE(0,1)= 17:COURSE(0,2)= 15:COURSE(0,3)=-17:
COURSE(2,0)= 15:COURSE(2,1)=-17:COURSE(2,2)=-15:COURSE(2,3)= 17
K__RED=-1:K_WHITE=1:K_KING__RED=-2:K_KINGWHITE=2:K__WALL=99:K_BLANK=0
'盤の周囲を壁で囲む
for I%=0 to 256:BOARD(I%)=K__WALL:next I%
while 1
                                         cls
                                         '駒を置く

BOARD(3*16+2)=K___RED:BOARD(5*16+2)=K___RED:BOARD(7*16+2)=K___RED:BOARD(9*16+2)=K___RED

BOARD(2*16+3)=K___RED:BOARD(4*16+3)=K___RED:BOARD(6*16+3)=K___RED:BOARD(8*16+3)=K___RED

BOARD(3*16+4)=K___RED:BOARD(5*16+4)=K___RED:BOARD(7*16+4)=K___RED:BOARD(9*16+4)=K___RED

BOARD(2*16+5)=K_BLANK:BOARD(4*16+5)=K_BLANK:BOARD(6*16+5)=K_BLANK:BOARD(8*16+5)=K_BLANK

BOARD(3*16+6)=K_BLANK:BOARD(5*16+6)=K_BLANK:BOARD(7*16+6)=K_BLANK:BOARD(9*16+6)=K_BLANK

BOARD(2*16+7)=K_WHITE:BOARD(4*16+7)=K_WHITE:BOARD(6*16+7)=K_WHITE:BOARD(8*16+7)=K_WHITE

BOARD(3*16+8)=K_WHITE:BOARD(5*16+8)=K_WHITE:BOARD(7*16+8)=K_WHITE:BOARD(9*16+8)=K_WHITE

BOARD(2*16+9)=K_WHITE:BOARD(4*16+9)=K_WHITE:BOARD(6*16+9)=K_WHITE:BOARD(8*16+9)=K_WHITE
                                         DISPLAY:WIN_LOSE_CHECK=0
                                         while 1
                                          WIN_LOSE_CHECK=PLAYER(1)
                                          DISPLAY:beep
                                          if WIN_LOSE_CHECK<>0 then goto *ENDING
                                          WIN_LOSE_CHECK=COMPUTER(-1)
                                          DISPLAY:beep
                                          if WIN_LOSE_CHECK<>0 then goto *ENDING
                                         wend
                                         *ENDING
                                         if WIN_LOSE_CHECK=1 then locate 50,3:print "私の負け
です"
                                         if WIN_LOSE_CHECK=2 then locate 50,3:print "あなたの
負け"
                                         locate 50,4:print "もう一度やりますか"
                                         locate 50,5:print "（Y左・N右）クリック"
                                         if MOUSEWAIT=1 then goto *BREAK
wend
*BREAK
KING__RED.DELETEOBJECT:KINGWHITE.DELETEOBJECT:BLANK.DELETEOBJECT:RED__.DELETEOBJECT
WHITE.DELETEOBJECT:WALL_.DELETEOBJECT:end
'盤と駒を表示
sub DISPLAY()
var I%,J%
```

★影さん、「網走・シベリア・バスチーユの旅」はいかがですか？キャンセル不可。（群馬県高崎市・巽 洋平22歳）……【影：招待してくれるなら、何の旅でも、どこでも行くぞ。つぐ美：バスチーユってどこにあるか知ってるの？影：だから海外のほう…】

入力方法は31ページをご覧ください。

F-BASIC

★あっ!3Dモデルやってる♪HSPのページ参考になりそう。でもfvdirとか便利そうで、そこんとこ考えなきゃダメかも。球の表面を移動させたいですよ。あっ、答えは言わないで!考えてるのが楽しいから。(愛知県名古屋市・わいち301, 27歳)……【影:えらい!】

```
                                        for I%=2 to 9:for J%=2 to 9
                                         if BOARD(I%*16+J%)= K__WALL      then DRAWBITMAP
WALL_    ,32*I%+4,32*J%+4
                                         if BOARD(I%*16+J%)= K___RED      then DRAWBITMAP
RED__    ,32*I%+4,32*J%+4
                                         if BOARD(I%*16+J%)= K_KINGWHITE then DRAWBITMAP
KINGWHITE,32*I%+4,32*J%+4
                                         if BOARD(I%*16+J%)= K_KING__RED then DRAWBITMAP
KING__RED,32*I%+4,32*J%+4
                                         if BOARD(I%*16+J%)= K_WHITE      then DRAWBITMAP
WHITE    ,32*I%+4,32*J%+4
                                         if BOARD(I%*16+J%)= K_BLANK      then DRAWBITMAP
BLANK    ,32*I%+4,32*J%+4
                                        next J%:next I%
end sub

function COMPUTER(COL%)
var OLD%,X%,Y%,I%,J%,MOVE_CHECK%,AMG%,CAPTURE_SP%,MOUSECHECK%,WIN_LOSE%
                                        EV2%=-
30000:MOVE_CHECK%=0:CAPTURE_SP%=0:AMG%=0:SP=0:WIN_LOSE%=0
                                        for OLD%=0 to 256
                                         if BOARD(OLD%)=COL% or BOARD(OLD%)=COL%*2 then
J%=COM_CAPTURE(COL%,OLD%,EV2%)
                                         if J%=2 then MOVE_CHECK%=2
                                        next OLD%
                                        if MOVE_CHECK%<>2 then goto *MOVE
                                        locate 50,15:print "sp1";SP
                                        SP=cint(rnd(1)*SP)
                                        locate 50,16:print "sp2";SP
                                        CAPTURE_SP%=MAX_CAPTURE_SP(SP)
                                        for I%=0 to CAPTURE_SP%
LST%=R_LST(I%,SP):NEW%=R_NEW(I%,SP):OLD%=R_OLD(I%,SP)
                                         AMG%=R_AMG(I%,SP):PRM%=R_PRM(I%,SP)
                                         BOARD(NEW%)=BOARD(OLD%):BOARD(OLD%)=0:BOARD(AMG%)=0
                                         if PRM%=1 then BOARD(NEW%)=COL%*2
                                         DISPLAY
                                        next I%
                                        if MOVE_CHECK%=2 then goto *RET1
*MOVE
                                        for OLD%=0 to 256
                                         if BOARD(OLD%)=COL% or BOARD(OLD%)=COL%*2 then
J%=COM_MOVE(COL%,OLD%,EV2%)
                                         if J%=1 then MOVE_CHECK%=1
                                        next OLD%
                                        if MOVE_CHECK%<>1 then goto *RET1
                                        locate 50,15:print "sp1";SP
SP=cint(rnd(1)*SP):NEW%=NNEW(SP):OLD%=OOLD(SP):PRM%=PRMT(SP)
                                        BOARD(NEW%)=BOARD(OLD%):BOARD(OLD%)=0
                                        if PRM%=1 then BOARD(NEW%)=COL%*2
                                        locate 50,16:print "sp2";SP
*RET1
```

```
                                        '狐が動けなくなったらコンピュータの負け
                                        if MOVE_CHECK%=0 then COMPUTER=1
*RET:
end function

function COM_NEW_CHECK(COL%,I%,OLD%,LST%)
var J%
                                        J%=COURSE(COL%+1,LST%):if I%=1 then J%=J%*2
                                        COM_NEW_CHECK=OLD%+J%
end function

function COM_CLIST_CHECK(COL%,I%,OLD%,LST%)
var CHECK%
                                        CHECK%=0
                                        if LST%=0 then goto *LIST0
                                        if LST%=1 then goto *LIST1
                                        if LST%=2 then goto *LIST2
                                        if LST%=3 then goto *LIST3
                                        LST%=100:goto *RET
*LIST0:                                 if I%=1 and BOARD(OLD%+COURSE(COL%+1,0))=-COL%   and
BOARD(OLD%+COURSE(COL%+1,0)*2)=0 then LST%=0:goto *RET
                                        if I%=1 and BOARD(OLD%+COURSE(COL%+1,0))=-COL%*2 and
BOARD(OLD%+COURSE(COL%+1,0)*2)=0 then LST%=0:goto *RET
                                        if I%=0 and BOARD(OLD%+COURSE(COL%+1,0))=0
then LST%=0:goto *RET
*LIST1:                                 if I%=1 and BOARD(OLD%+COURSE(COL%+1,1))=-COL%   and
BOARD(OLD%+COURSE(COL%+1,1)*2)=0 then LST%=1:goto *RET
                                        if I%=1 and BOARD(OLD%+COURSE(COL%+1,1))=-COL%*2 and
BOARD(OLD%+COURSE(COL%+1,1)*2)=0 then LST%=1:goto *RET
                                        if I%=0 and BOARD(OLD%+COURSE(COL%+1,1))=0
then LST%=1:goto *RET
*LIST2: if BOARD(OLD%)=COL% then LST%=100:goto *RET
                                        if I%=1 and BOARD(OLD%+COURSE(COL%+1,2))=-COL%   and
BOARD(OLD%+COURSE(COL%+1,2)*2)=0 then LST%=2:goto *RET
                                        if I%=1 and BOARD(OLD%+COURSE(COL%+1,2))=-COL%*2 and
BOARD(OLD%+COURSE(COL%+1,2)*2)=0 then LST%=2:goto *RET
                                        if I%=0 and BOARD(OLD%+COURSE(COL%+1,2))=0
then LST%=2:goto *RET
*LIST3: if BOARD(OLD%)=COL% then LST%=100:goto *RET
                                        if I%=1 and BOARD(OLD%+COURSE(COL%+1,3))=-COL%   and
BOARD(OLD%+COURSE(COL%+1,3)*2)=0 then LST%=3:goto *RET
                                        if I%=1 and BOARD(OLD%+COURSE(COL%+1,3))=-COL%*2 and
BOARD(OLD%+COURSE(COL%+1,3)*2)=0 then LST%=3:goto *RET
                                        if I%=0 and BOARD(OLD%+COURSE(COL%+1,3))=0
then LST%=3:goto *RET
                                        LST%=100
*RET
                                        COM_CLIST_CHECK=LST%
end function

function PLAYER(COL%)
var X%,Y%,OLD%,NEW%,AMG%,CHECK%,MOVE_CHECK%,KOMA%,PRM%
```

```
                              locate 50,1:print "あなたの番です"
                              EV2%=-30000:COM_MOVE_CHECK%=0:SP=0
                              for OLD%=0 to 256
                               if BOARD(OLD%)=COL% or BOARD(OLD%)=COL%*2 then
J%=COM_CAPTURE(COL%,OLD%,EV2%)
                               if J%=2 then COM_MOVE_CHECK%=2
                              next OLD%
                              if COM_MOVE_CHECK%=2 then goto *JUMP0
                              for OLD%=0 to 256
                               if BOARD(OLD%)=COL% or BOARD(OLD%)=COL%*2 then
J%=COM_MOVE(COL%,OLD%,EV2%)
                               if J%=1 then COM_MOVE_CHECK%=1
                              next OLD%
                              if COM_MOVE_CHECK%=0 then PLAYER=2:goto *RET
*JUMP0
                              if COM_MOVE_CHECK%=2 then locate 50,2:print "相手の駒
を取って下さい"
*LOOP0
                              MOVE_CHECK%=0
                              while MOVE_CHECK%=0
                               OLD%=MAN_INPUT_CHECK(1)
                               KOMA%=BOARD(OLD%)
                               NEW%=MAN_INPUT_CHECK(2)
                               if COM_MOVE_CHECK%=2 then
MOVE_CHECK%=MAN_JUMP_CHECK(COL%,OLD%,NEW%)
                               if COM_MOVE_CHECK%=1 then
MOVE_CHECK%=MAN_MOVE_CHECK(COL%,OLD%,NEW%)
                               if COM_MOVE_CHECK%=2 and MOVE_CHECK%<>2 then locate
50,3:print "そこには進めません。":X%=(OLD% ¥ 16):Y%=(OLD% mod 16):line(X%*32+3,Y%*32+3)-
step(32,32),pset,9,b
                               if COM_MOVE_CHECK%=1 and MOVE_CHECK%<>1 then locate
50,3:print "そこには進めません。":X%=(OLD% ¥ 16):Y%=(OLD% mod 16):line(X%*32+3,Y%*32+3)-
step(32,32),pset,9,b
                              wend
                              AMG%=int((OLD%+NEW%)/2)
                              PRM%=PROMOTION_CHECK(COL%,OLD%,NEW%)
                              if MOVE_CHECK%=2 then
BOARD(OLD%)=0:BOARD(AMG%)=0:BOARD(NEW%)=KOMA%
                              if MOVE_CHECK%=1 then BOARD(OLD%)=0:
BOARD(NEW%)=KOMA%:goto *RET
                              if PRM%=1 then BOARD(NEW%)=COL%*2:goto *RET
DISPLAY:OLD%=NEW%:COM_MOVE_CHECK%=COM_CAPTURE(COL%,OLD%,EV2%)
                              if COM_MOVE_CHECK%=2 then locate 50,2:print "もう一つ
獲ってください":goto *LOOP0
*RET
                              locate 50,1:print "                    "
                              locate 50,2:print "                    "
                              locate 50,3:print "                    "
                              if PRM%=1 then BOARD(NEW%)=COL%*2
end function

function MAN_INPUT_CHECK(IN_OR_OUT%)
var CHECK%,CELL%
                              while CHECK%=0
                               MOUSECHECK=MOUSEWAIT
                               X%=int(mouse(0)/32):Y%=int(mouse(1)/32)
                               CELL%=X%*16+Y%
                               if CELL%<0 and CELL%>255 then goto *JUMP
                               if IN_OR_OUT%=2 then CHECK%=1:goto *JUMP
                               if BOARD(CELL%)>0 and BOARD(CELL%)<17 then
CHECK%=1:line(X%*32+3,Y%*32+3)-step(32,32),pset,5,b else locate 50,3:print "其処に駒はあり
ません"
                               *JUMP
                              wend:MAN_INPUT_CHECK=CELL%
end function

function MAN_JUMP_CHECK(COL%,OLD%,NEW%)
var CHECK%
                              MAN_JUMP_CHECK=0
                              if BOARD(NEW%)<>0 then goto *RET
                              if NEW%=OLD%+COURSE(COL%+1,0)*2 then
MAN_JUMP_CHECK=2:'左前
                              if NEW%=OLD%+COURSE(COL%+1,1)*2 then
MAN_JUMP_CHECK=2:'右前
                              if BOARD(OLD%)=COL% then goto *RET
                              if NEW%=OLD%+COURSE(COL%+1,2)*2 then
MAN_JUMP_CHECK=2:'左後
                              if NEW%=OLD%+COURSE(COL%+1,3)*2 then
MAN_JUMP_CHECK=2:'右後
*RET
end function

function MAN_MOVE_CHECK(COL%,OLD%,NEW%)
var CHECK%
                              MAN_MOVE_CHECK=0
                              if BOARD(NEW%)<>0 then goto *RET
                              if NEW%=OLD%+COURSE(COL%+1,0)  then
MAN_MOVE_CHECK=1:'左前
                              if NEW%=OLD%+COURSE(COL%+1,1)  then
MAN_MOVE_CHECK=1:'右前
                              if BOARD(OLD%)=COL% then goto *RET
                              if NEW%=OLD%+COURSE(COL%+1,2)  then
MAN_MOVE_CHECK=1:'左後
                              if NEW%=OLD%+COURSE(COL%+1,3)  then
MAN_MOVE_CHECK=1:'右後
*RET
end function

'右クリックの時1を左クリックの時0を還す
function MOUSEWAIT()
                              while mouse(2,1)<>-1 and mouse(2,0)<>-1
                               if mouse(2,1)=-1 then MOUSEWAIT=1 else  MOUSEWAIT=0
                              wend:while mouse(2,1)=-1 or mouse(2,0)=-1:wend
end function
```

★図らずも本誌イラスト投稿の常連さんが寄稿された同*誌に遭遇。世の中せまいとみるべきか、人はどこかでつながっていると見るべきか。(東京都狛江市・**星屑20歳**)……**【影：せまいから，つながってるんでしょうね。つぐ美：影さんが言うと夢がこわれちゃう。】**

F-BASIC

入力方法は31ページをご覧ください。

★特（性）Tシャツ（3月号126ページ）って、どんな特質のTシャツなですか？（茨城県古河市・**年齢不相応42歳**）……**【影：**AIBOだから特性なの。合金でねー。**つぐ美：**読者のみなさんすみません。誤字でした。ほら影さんも。**影：**だからピカピカの…ごめんなさい。**】**

```
'##########################################################################
'# 連続キャプチャーを調べるアルゴリズムです。一番多く相手の駒を取れる手を指します。
'# この部分が一番苦労しました。色々コメントを付けていますが、自分でも処理の流れを追って
'# いて頭が痛くなってきます。
'##########################################################################
function COM_CAPTURE(COL%,OLD%,EV2%)
var X%,Y%,I%,J%,MOVE_CHECK%,AMG%,CAPTURE_SP%,PRM%,ENM%
                                   '変数を初期化
                                   MOVE_CHECK%=0:CAPTURE_SP%=0:AMG%=0:MOVE_CHECK%=0
*LOOP1
                                   LST%=COM_CLIST_CHECK(COL%,1,OLD%,0)
                                   '捕獲出来ない時は普通の移動をする
                                   if MOVE_CHECK%=0 and LST%>10 then goto *ret
                                   '捕獲出来る時はチェックをする
                                   if MOVE_CHECK%=0 and LST%<10 then MOVE_CHECK%=2:goto
*JUMP2
                                   '連続捕獲が終わったらループから抜ける
                                   if MOVE_CHECK%=2 and LST%>10 then goto *BREAK
*JUMP2
                                   NEW%=COM_NEW_CHECK(COL%,1,OLD%,LST%)
                                   PRM%=PROMOTION_CHECK(COL%,OLD%,NEW%)
                                   AMG%=int(OLD%+NEW%)/2
                                   LIST(CAPTURE_SP%)=LST%:NNEW(CAPTURE_SP%)=NEW%
                                   OOLD(CAPTURE_SP%)=OLD%:AMNG(CAPTURE_SP%)=AMG%
                                   ENMY(CAPTURE_SP%)=BOARD(AMG%)
                                   PRMT(CAPTURE_SP%)=PRM%
                                   CAPTURE_SP%=CAPTURE_SP%+1
                                   BOARD(NEW%)=BOARD(OLD%):BOARD(AMG%)=0:BOARD(OLD%)=0
'@@@@@@@@@@@@@@@@@@@@ ＢＵＧ獲り @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
                                   if PRM%=1 then BOARD(NEW%)=COL%*2:goto
*BREAK:'DISPLAY
                                   'DISPLAY
                                   OLD%=NEW%
                                   goto *LOOP1
*BREAK
                                   CAPTURE_SP%=CAPTURE_SP%-1
                                   if CAPTURE_SP%<0 then goto *RET
                                   '捕獲する手筋を記録していく
                                   EV1%=CAPTURE_SP%
                                   if EV1%<EV2% then goto *LOOP2
                                   if EV1%>EV2% then SP=0:EV2%=EV1%:goto *JUMP11
                                   SP=SP+1
*JUMP11
                                   MAX_CAPTURE_SP(SP)=CAPTURE_SP%
                                   for I%=0 to CAPTURE_SP%

R_LST(I%,SP)=LIST(I%):R_NEW(I%,SP)=NNEW(I%):R_OLD(I%,SP)=OOLD(I%)

R_AMG(I%,SP)=AMNG(I%):R_PRM(I%,SP)=PRMT(I%):R_ENM(I%,SP)=ENMY(I%)
                                   next I%
                                   '手を戻していくルーチン
                                   *LOOP2
```

```
LST%=LIST(CAPTURE_SP%):NEW%=NNEW(CAPTURE_SP%):OLD%=OOLD(CAPTURE_SP%)

AMG%=AMNG(CAPTURE_SP%):PRM%=PRMT(CAPTURE_SP%):ENM%=ENMY(CAPTURE_SP%)
                                   '駒を一手戻す

BOARD(OLD%)=BOARD(NEW%):BOARD(NEW%)=0:BOARD(AMG%)=ENM%
                                   if PRM%=1 then BOARD(OLD%)=COL%
'@@@@@@@@@@@@@@@@@@@@ ＢＵＧ獲り @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
'                                  DISPLAY:wait 50
                                   '別の手を探す
                                   LST%=LST%+1:LST%=COM_CLIST_CHECK(COL%,1,OLD%,LST%)
                                   if LST%<10 then goto *JUMP2
                                   '別の手が無いときはさらに一手戻す
                                   CAPTURE_SP%=CAPTURE_SP%-1:if CAPTURE_SP%<0 then goto
*RET
                                   goto *LOOP2
*RET
                                   COM_CAPTURE=MOVE_CHECK%
end function

function PROMOTION_CHECK(COL%,OLD%,NEW%)
var Y%
                                   Y%=NEW% mod 16:PROMOTION_CHECK=0
                                   if COL%=-1 and BOARD(OLD%)=COL% and Y%=9 then
PROMOTION_CHECK=1
                                   if COL%= 1 and BOARD(OLD%)=COL% and Y%=2 then
PROMOTION_CHECK=1
end function

'普通のムーブ
function COM_MOVE(COL%,OLD%,EV2%)
var LST%,EV1%,CHECK%,PRM%,NEW%
                                   CHECK%=0:LST%=0
                                   while 1
                                   EV1%=0:LST%=COM_CLIST_CHECK(COL%,0,OLD%,LST%)
                                   if LST%>10 then goto *RET

NEW%=COM_NEW_CHECK(COL%,0,OLD%,LST%):PRM%=PROMOTION_CHECK(COL%,OLD%,NEW%)
                                             LST%=LST%+1
                                             if EV1%<EV2% then goto *JUMP
                                             if EV1%>EV2% then SP=0
:NNEW(SP)=NEW%:OOLD(SP)=OLD%:PRMT(SP)=PRM%:EV2%=EV1%:CHECK%=1:goto *JUMP

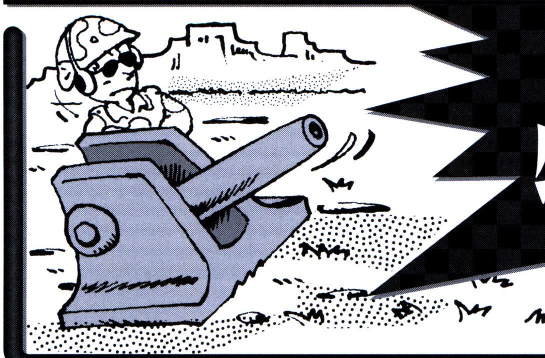
SP=SP+1:NNEW(SP)=NEW%:OOLD(SP)=OLD%:PRMT(SP)=PRM%
                                   *JUMP
                                   wend
*RET
                                   COM_MOVE=CHECK%
end function
```

Windows(HSP V2.5)用

ポケコンゲームをHSPでアレンジ

移植版

物体捕獲ゲーム



三好　利治（46歳）

入力方法は31ページをご覧ください。

はじめに

本誌2002年１月号掲載のポケコン(PC-G850)用に作られた関谷様作の『TARGET_B』を，HSP2.5用へ移植したものです(**写真１**)。

ただ，実際にポケコンでの動作を確認していませんので，内容を想像しての移植です。

使用方法

使用するキーは，ゲームを始めるときなどに使用するスペースキーと，ワクを動かすカーソルキーです。

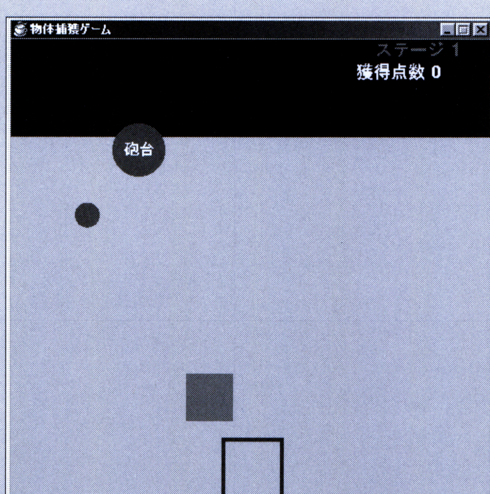
ワクを移動させ，逃げる物体に重ねて捕獲します(**写真２**)。ワクは，← → ↑ ↓キーにて移動します。

砲台(弾出しキャラ)が弾を撃ってきます。弾に当るとゲームオーバーです(**写真３**)。

ステージが進むほど，難易度が上がります(被捕獲物体の挙動の変更や，その物体の移動スピード，弾の数，弾の移動スピードが変化します)。

ステージが進むほど獲得点数が高くなります。早く捕まえると(20，40，60秒以内)ボーナスが加算されます。

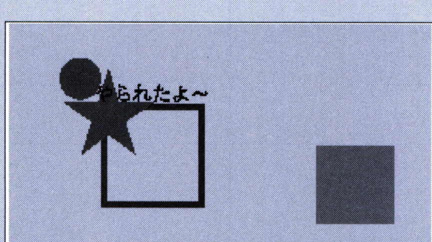
非常に単純な内容のゲームですが，イ



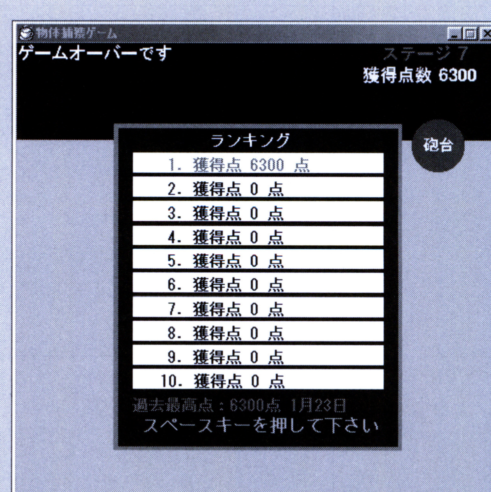
《写真１》動き回る物体を捕獲するゲーム



《写真２》ワクの中に物体を入れればＯＫ



《写真３》砲台の弾にぶつかるとダメ



《写真４》ハイスコアのランキング機能もある

★小向ちゃんを,気に入った。（千葉県山武郡・**野潤上15歳**）……【**編**：たまに，OFコーナーを告白の場にする人がいますが，恋が芽生えたことってあるのでしょうか。**影**：ところで，小向ちゃんって誰？**編**：さあ？**つぐ美**：がんばって！**編**：何を？**つぐ美**：さあ？】

《リスト》物体捕獲ゲームのプログラム・リスト

```
;***** 捕獲ゲーム hokaku.as HSP2.5 2002/1 toshi.miyoshi

        screen 0,500,480,1,dispx-500/2,dispy-480/2 ;256色モード

        sndload "ok.wav",2       ;効果音
        sndload "kan.wav",3

        dim rank,11          ;ランキング用配列

*saido                                   ;初期画面(説明画面)
        stg=1        ;ステージ数
        tenngou=0    ;点数合計

        title "物体捕獲ゲーム"
        color 192,220,192:boxf

        font "MSゴシック",30,1
        color 0,0,0
        pos 120,12:mes "<物体捕獲ゲーム>"
        color 255,255,255
        pos 118,10:mes "<物体捕獲ゲーム>"

        color 0,0,255     :boxf 20,50,20+64,50+64
        color 192,220,192:boxf 20+4,50+4,20+60,50+60
        font "MSゴシック",20,1 :color 0,0,0
        pos 86,80:mes "が、あなたです"
        color 255,0,0     :boxf 225+8,50+8,225+56,50+56
        color 0,0,0
        pos 290,80:mes "を捕獲するゲームです"
        pos 40,120:mes "矢印キーで動かし、物体を捕らえてください"
        pos 40,145:mes "←・→・↑・↓  です  同時で斜めに動きます"
        color 0,128,0:font "MSゴシック",64,1
      pos 20,170:mes "●"
        color 255,255,0:font "MSゴシック",16,1
        pos 20+16,170+22:mes "砲台"
        font "MSゴシック",20,1 :color 0,0,0
        pos 86,200:mes "の出す弾に当れば、ゲームオーバーです"
        pos 40,250:mes "ステージが進むほど、難易度が上がります"
        pos 40,275:mes "ステージが進むほど獲得点数が高くなります"
        pos 40,300:mes "早く捕まえると、ボーナスが加算されます"

        color 128,0,0
        pos 40,400 :mes "では、スペースキーを押すと始まります。"
        pos 350,440:mes "【Esc】:終了"
*tloop
        wait 1:stick ky,0:if ky=128:end :Escで終了
        if ky!16:goto *tloop

*hajime                                  :ステージクリアごとにここへ戻る

;捕獲者の初期設定
        px=220 :py=410                   :初期位置
        hp=1                             ;耐久値(一回当れば終わり,拡張可)
```

ライラすること請け合いです。

ちなみに斜めに動かすと，捕獲(重ね合わせ)しやすいです。また，被捕獲物体の動きによっては，勝手に捕獲できることもあります。

画像などは使用していませんので，リ

入力方法

ストを打ち込むだけで動作します。

保存用データファイルも，自動的に作成されますので，用意する必要がありません。

効果音ファイルは，「カシャン」「バシッ」などといった感じの音を，サウンドレコーダーで録音するなどの手段で用意し，ok.wav , kan.wav　というファイル名で保存してください。

HSPは，2002年１月号のベストプロ

プログラムの仕組み

グラム研究室でDr.Dが解説していた，VBのようなLavelを使った移動，捕獲判定，弾の当たり判定が出来ません。そこで，当たり判定などは，昔ながらの各キャラクタの座標位置を用いた形で行っています。

自分の移動量は，キーを押すごとに８ドット移動するようにしました(それ以下にすると，とても遅くなりますので)。

捕獲判定は，物体と座標がピッタリ合ったときだけでなく，ある程度の幅をもたせて判定しています。その結果，磁石のような効果が出ました(ちょっと苦し紛れですね)。

キャラクタに画像を利用すると，きれ

改造など

いなものが出来ると思います。

リスト上から変数の値を変更して，プレイヤーの好みに合わせて，難易度を調整することも可能です。

また，拡張用に被捕獲物体の数，砲台の数を変数(配列のインデックス)にしてありますので，その値を変えるだけで，簡単に敵の数を複数に出来ます(リストではrepeat-loopの回数にしてあります)。

CHECKER FLAG

編：『TARGET_B』の移植版という形ですが，原作をプレイした側からいうと，これはまったく違う形態のゲームと言えます。ですが，基本ルールは同じなので，しいて言えば『TARGET_B』のアレンジ版ですかね。

つぐ美：原作とは異なっているけど，これはこれでなかなか面白くできているゲームですよ。スピード感があるゲームに仕上がっていますね。

Dr.D:多少なりともワシのコーナーを参考にしてくれたのかな？そうであればウレシイ限りじゃな。形態は違えど，アルゴリズムは大きく変わってはいない。原作のアルゴリズムは，説明文にもあるように１月号のベスプロ研究室で解説しているので，ぜひそちらにも目を通してみてほしい。

★本体同様，最近イマイチ元気がない「WonderWitch」の投稿。こうなったら自分で投稿だ。(神奈川県横浜市・バビブ改めボーちゃん20歳)……【**影**：掛け声だけじゃなく，本当に投稿してくれるんだろうな。**編**：ちなみに今月は，畑の木さんの投稿が掲載されてます。】

```
;被捕獲者と砲台(弾発射キャラ)の初期設定
	randomize
	eemx=1                         ;被捕獲者の数(1個,拡張可)
	dim eex,eemx :dim eey,eemx
	dim ddx,eemx :dim ddy,eemx
	repeat eemx
	  rnd eex.cnt,420 :rnd eey.cnt,410 ;被捕獲者の位置
	  if eey.cnt<100:eey.cnt=100
	  rnd r,5 :ddx.cnt=r-2*2 :rnd r,5 :ddy.cnt=r-2*2
	loop

	emx=1                          ;砲台(弾発射キャラ)の数(1匹,拡張可)
	ecn=emx                        ; "" チェック用
	dim ehp,emx                    ; "" 耐久値
	dim ex,emx :dim ey,emx         ; "" の位置
	dim dx,emx :dim dy,emx         ; "" の移動方向
	repeat emx           ; "" の数だけループ
	  ehp.cnt=1
	  rnd ex.cnt,450 :rnd ey.cnt,100             ;キャラ位置
	  rnd r,5 :dx.cnt=r-2*2 :rnd r,5 :dy.cnt=r-2*2    ;移動方向
	loop

	;難易度設定(弾の数、速さ、被捕獲者の動く速さ)
	;ermx:弾数  ttr:動く頻度  tti:動きの速さ  tamai:弾の速さ
	;自分の好みに変えて下さい(配列変数にするといいと思います)
	if stg=1:etmx=1:ttr=100:tti=2:tamai=2
	if stg=2:etmx=2:ttr=100:tti=2:tamai=2
	if stg=3:etmx=3:ttr=100:tti=2:tamai=2
	if stg=4:etmx=2:ttr=50:tti=3:tamai=3
	if stg=5:etmx=3:ttr=50:tti=3:tamai=3
	if stg=6:etmx=4:ttr=50:tti=3:tamai=3
	if stg=7:etmx=3:ttr=10:tti=5:tamai=4
	if stg=8:etmx=4:ttr=10:tti=5:tamai=4
	if stg=9:etmx=5:ttr=10:tti=5:tamai=4
	if stg=10:etmx=5:ttr=5:tti=7:tamai=5
	if stg>=11:etmx=7:ttr=5:tti=7:tamai=5

	dim etf,etmx                        ;弾フラグ
	dim etx,etmx :dim ety,etmx     ;弾の位置、
	dim tdx,etmx :dim tdy,etmx     ;弾移動方向の配列
	repeat etmx                     ;弾発射フラグのクリア
	  etf.cnt=0
	loop

;ゲーム進行メインルーチン
	gosub *tgettime1  ;ワンステージの時間開始取得
*main
	redraw 2            ;画面に反映しない(仮想画面は書き換わる)
	color 0,0,0:boxf
	color 192,220,192:boxf 0,100,500,480
	font "MSゴシック",18,1:color 255,0,0
	pos 380,2:mes "ステージ "+stg
	color 255,255,0
	pos 360,24:mes "獲得点数 "+tenngou

	await 1
	stick ky,15                                  ;カーソルの入力
	if ky&1 :px-=8 :if px<8 :px=8              ;自キャラの移動
	if ky&4 :px+=8 :if px>436 :px=436          ;  斜めも可
	if ky&2 :py-=8 :if py<100 :py=100          ;  同時に押す
	if ky&8 :py+=8 :if py>416 :py=416          ;1キー押下8ドット
	color 0,0,255    :boxf px,py,px+64,py+64    ;この位置に自分を表示
	color 192,220,192:boxf px+4,py+4,px+60,py+60

	repeat eemx          ;被捕獲者の動き
	  rnd r,ttr      ;動きの変化ttr  動く距離tti
	  if r=0 :rnd r,5 :ddx.cnt=r-2*tti :rnd r,5 :ddy.cnt=r-2*tti
	  x=eex.cnt :eex.cnt+=ddx.cnt
	  if (eex.cnt<14)|(eex.cnt>420)  :eex.cnt=x :ddx.cnt=-ddx.cnt
	  y=eey.cnt :eey.cnt+=ddy.cnt
	  if (eey.cnt<100)|(eey.cnt>410) :eey.cnt=y :ddy.cnt=-ddy.cnt
	  color 255,0,0                 ;この位置に被捕獲者を表示
	  boxf eex.cnt+8,eey.cnt+8,eex.cnt+56,eey.cnt+56
	loop

;捕獲者と被捕獲者が重なった判定(磁石効果あり^^;)

	tx1=px+8:tx2=px-8:ty1=py+8:ty2=py-8
	if (eex.cnt<=tx1)&(eex.cnt>=tx2)&(eey.cnt<=ty1)&(eey.cnt>=ty2){
	    snd 2 : ttx=eex.cnt:tty=eey.cnt
	    color 192,220,192:boxf ttx-4,tty-4,ttx+68,tty+68
	    ;捕獲時の表示
	    color 0,0,255    :boxf px,py,px+64,py+64
	    color 192,220,192:boxf px+4,py+4,px+60,py+60
	    color 255,0,0    :boxf px+8,py+8,px+56,py+56
	    color 255,255,0:font "MSゴシック",20,1
	    pos px+12,py+12:mes "捕獲":mes "成功"
	    hokaku=1                          ;捕獲フラグオン
	    gosub *tgettime2                  ;捕獲までの終了時間取得
	    goto *hskip                       ;弾に当らないようスキップ
	}

	repeat emx                      ;砲台(弾出しキャラ)の動き
	  rnd r,5
	  if r=0 :rnd r,5 :dx.cnt=r-2*4 :rnd r,5 :dy.cnt=r-2*2
	x=ex.cnt :ex.cnt+=dx.cnt
	  if (ex.cnt<8)|(ex.cnt>430) :ex.cnt=x :dx.cnt=-dx.cnt
	  y=ey.cnt :ey.cnt+=dy.cnt
	  if (ey.cnt<8)|(ey.cnt>100) :ey.cnt=y :dy.cnt=-dy.cnt
	  color 0,128,0:font "MSゴシック",64,1
	pos ex.cnt,ey.cnt : mes "●"          ;この位置に表示
	  color 255,255,0:font "MSゴシック",16,1
	  pos ex.cnt+16,ey.cnt+22 : mes "砲台"
	loop

	rnd r,etmx :rnd e,emx          ;弾の発射  弾の速さtamai
	if (etf.r=0)&(ehp.e>0) {
```

★春はあこぼの。夏は夜。秋は焼き芋。冬はこたつみかん。(長野県更埴市・たこ12歳)……【編:これに応えて影さんが一句。影:え?では。春はカゲ。夏もカゲ。秋はツグミ。冬はヘン。編:なんかへん。影:頭の回転がいまいちなんだよ。つぐ美:いつもね。】

HSP

入力方法は31ページをご覧ください。

★ソルトレイクシティオリンピックが行われている中，ファミコンのバルセロナ192で盛り上がっている自分が悲しいです。（愛媛県東宇和郡・**毛利祐允18歳**）……【**つぐ美**：影さんゲーム・センターね。**編**：なんでも懐かしの「ハイパーオリンピック」見つけたとかで。】

```
            etx.r=ex.e+4 :ety.r=ey.e+16
            rnd x,3 :tdx.r=x-2*tamai :rnd y,4 :tdy.r=y+1*tamai
            etf.r=1
        }
        repeat etmx                         ;弾の数、弾との当たり判定
          x=etx.cnt :y=ety.cnt
          if (etf.cnt=1)&(px+32>x)&(x+32>px)&(py+32>y)&(y+32>py) {
              htc=2 :hx=etx.cnt :hy=ety.cnt :snd 3
              etf.cnt=0 :hp- :if hp<1 :bmc=3 :bmx=px :bmy=py
          }
          if etf.cnt=1 {
              color 0,128,0:font "MSゴシック",32
              pos etx.cnt,ety.cnt : mes "●"        ;弾の表示
              etx.cnt+=tdx.cnt :ety.cnt+=tdy.cnt
              if (etx.cnt<0)|(etx.cnt>468)|(ety.cnt>448) :etf.cnt=0
          }
        loop
                                            ;命中時の火花表示
        if htc>0 {
           htc-
           color 0,128,0:font "MSゴシック",64
           pos hx,hy : mes "★"
        }
*hskip
        redraw 1                    ;画面に反映させる

        if hokaku=1{                ;捕獲成功処理
           gosub *tosimes           ;ステージクリア点数表示
           goto *skip2              ;次のステージへ進む
        }

        if hp<=0{                   ;ゲームオーバ
           gosub *tosimes2          ;やられた表示
           gosub *tkekka            ;結果ランキング表示
           goto *saido              ;初期画面へ
      }

        if hp>0  :goto *main        ;弾に当らず捕獲も出来なかったので
                                            ;同じステージを繰返す
*skip2
        wait 50
        font "MSゴシック",18,1:color 0,255,0
        pos 0,22:mes "スペースキーを押してください"

*tloop2
        wait 1:stick ky,0:if ky!16:goto *tloop2
        goto *hajime  ;ステージアップして続ける

;=========サブルーチン==============

*tosimes
        tenn=stg*100                        ;基本点はステージ数X１００点
        bonas=0                             ;ボーナス点(下記時間に応じて加算)
        font "MSゴシック",12,1
        color 0,128,0
```

```
        if tk>60 :pos px,py-24:mes "ﾎﾞｰﾅｽ無"
        if (tk>40)&(tk<=60){                ;60秒以内、小ボーナス獲得
           bonas=stg*50
           pos px,py-24:mes "小ﾎﾞｰﾅｽ付"
        }
        if (tk>20)&(tk<=40){                ;40秒以内、中ボーナス獲得
           bonas=stg*100
           pos px,py-24:mes "中ﾎﾞｰﾅｽ付"
        }
        if tk<=20{                          ;20秒以内、大ボーナス獲得
           bonas=stg*200
           pos px,py-24:mes "大ﾎﾞｰﾅｽ付"
        }

        pos px-8,py-12:mes " "+bonas+"点+"+tenn+"点"
        tenngou=tenngou+tenn+bonas

        color 0,0,0:boxf 360,24,480,42
        font "MSゴシック",18,1
        color 255,255,0
        pos 360,24:mes "獲得点数 "+tenngou
        color 255,255,255
        pos 0,2:mes "ステージ "+stg+" クリア 次進みましょう"
        stg=stg+1       ;ステージアップ
        hokaku=0        ;捕獲フラグオフ

        return


*tosimes2
        font "MSゴシック",12,1
        color 0,0,0
        pos px-4,py-12:mes "やられたよ～"
        font "MSゴシック",18,1
        color 255,255,255
        pos 0,2:mes "ゲームオーバーです"

        return

;時間測定計算
*tgettime1
        gettime tm1,4
        gettime tm2,5
        gettime tm3,6
        return
;
*tgettime2
        gettime etm1,4
        gettime etm2,5
        gettime etm3,6
        ts=tm1*3600:ts=tm2*60+ts:ts=tm3+ts
        te=etm1*3600:te=etm2*60+te:te=etm3+te
        tk=te-ts                            ;ワンステージ時間(秒)
        gtm=gtm+tk:secgou=secgou+tk         ;合計秒数加算(拡張用)
        return
```

```
;結果表示(ランキング表示とファイルへの更新など)
;別ウィンドウに表示する
*tkekka
        wait 100

        bgscr 2,300,340,1,dispx-300/2,dispy-340/2 ;評価画面
        gsel 2,1
        color 255,0,255:boxf
        color 0,0,128:boxf 5,5,295,335

        kousinn=0          ;ファイル更新、ハイスコア表示フラグ
        saikoubf="" : gappibf=""
                                    ;ファイルが存在するか？無ければ作成する
        exist "gappi.dat"
        if strsize=-1 : bsave "gappi.dat",gappibf
        exist "saikou.dat"
        if strsize=-1 : bsave "saikou.dat",saikoubf

        bload "gappi.dat",gappibf    ;過去最高点月日ファイルをロード
        bload "saikou.dat",saikoubf  ;過去最高得点ファイルをロード
        int saikoubf
        if tenngou > saikoubf :kousinn=1 :else: kousinn=0

        rank.10=tenngou  ;今回の点数をとりあえず最後に格納
        souto=0
        repeat 11
          if rank.cnt!0:souto=1
        loop
        if souto=0:goto *skip4  ;全て0点の場合はソートしない

;rank.0～rank.10をソートする:データ個数id=11 バブルソート

        id=11
        repeat id-1
          nct=cnt
            repeat id-nct-1
              j=cnt
              ne=j+1
              if (rank.j >= rank.ne):goto *noswp
        ;スワップ
              tmp=rank.j
              rank.j=rank.ne:rank.ne=tmp
*noswp
            loop
        loop

*skip4
        repeat 10                       ;10位以内か
          if tenngou=rank.cnt:jyun=cnt:break
        loop

        font "ＭＳ ゴシック",16,1:color 255,255,255
        pos 100,10:mes "ランキング"
        boxf 20,30,280,50
        boxf 20,55,280,75
        boxf 20,80,280,100
        boxf 20,106,280,125
        boxf 20,130,280,150
        boxf 20,155,280,175
        boxf 20,180,280,200
        boxf 20,205,280,225
        boxf 20,230,280,250
        boxf 20,255,280,275
        repeat 10                         ;今回結果は赤表示
          j=cnt+1
          if cnt=jyun:color 255,0,0:else:color 0,0,0
          pos 40,cnt*25+35
          if j=10{
             mes " "+j+". 獲得点 "+rank.cnt+" 点"
          }
          else{
             mes "  "+j+". 獲得点 "+rank.cnt+" 点"
          }
        loop

        color 255,0,0
        if kousinn=1{             ;過去最高をファイル更新
           gettime mmm,1:gettime ddd,3
           mmmm=mmm :dddd=ddd
           str mmmm :str dddd :mmdd=""
           mmdd=mmmm+"月"+dddd+"日"
           bsave "gappi.dat",mmdd
           kari=tenngou
           str kari
           bsave "saikou.dat",kari
           pos 20,285 :mes "新記録達成です "+tenngou+"点 "+mmdd
        }
        else{
           pos 20,285 :mes "過去最高点：  "+saikoubf+"点 "+gappibf
        }

        color 0,255,0
        font "ＭＳ ゴシック",18,1
        pos 30,305
        mes "スペースキーを押して下さい"
*tloop4
        wait 1:stick ky,0:if ky!16:goto *tloop4
        gsel 2,-1
        gsel 0,1
        return

;===============//EOD=====================
```

★最近、定期購読の振替用紙が付いていませんが、定期購読はできないんですか？（静岡県田方郡・**大車輪大助18歳**）……**【編：定期購読も受付中です。電話で直接お申し込みいただくか、メールでお知らせください。webmas@basicmagazine.net よろしく！】**

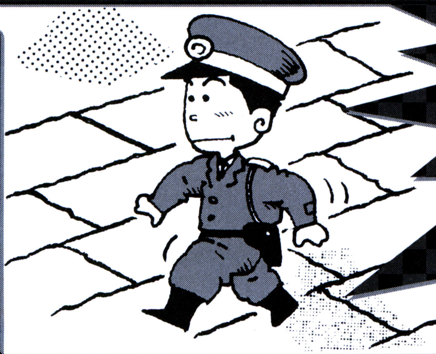
HSP

入力方法は31ページをご覧ください。

iモード対応携帯電話(iアプリ)用

あみだくじの原理で移動

アミダアミーゴ！



ヅラChu (19歳)

はじめに

例によって単純でハマるゲームを目指して作ってみました。F503i/D503i/SO503iで実機動作を確認しています。

遊び方

プレイヤーは，2，5，8と書かれた位置に，それぞれ，[2]，[5]，[8]キーを押すことで，アミダ線を引くことができます(**写真2**)。一応，上下・選択キーでも置けるようにしてあります。

自キャラ(泥棒)はアミダに沿って進むので，警官や警察犬に捕まらないよう，線を引いて誘導してください。警官は待ちかまえてるだけですが，警察犬はアミダを渡ってこちらに向かってくる(賢い！)ので，注意が必要です(**写真3**)。

警官と警察犬が衝突した場合，両者ダウンして，跡にはドクロマークが残ります。これは自キャラが捕まった時にも出てきますが，上を通っても問題ありません。

点数は前に1歩進むごとに1点，ドル袋を取ると100点もらえます。ハイスコア保存機能付き。

プログラムについて

amida.java(**リスト1**)をご使用の環境で入力してください。おいらの場合Java2SDK1.3とJ2MEWirelessSDKforDoJaに加えて，i-JADE Lite，西村誠一氏作『iアプリ開発Tool』(フリーソフト)を使っています。画像はすべて12×12ドットで作成し，それぞれ0.gif～4.gifで保存します。最後にプリベリファイ，jar圧縮して，jamファイル(**リスト2**)を作成すれば完成です。

プログラムについてですが，アミダを渡らせるところに苦労しました。自キャラと警察犬は動きがほぼ同じなので，無理矢理配列にしてみたりとか。ソースちょっと汚くなってるかも…（汗)。他は特筆すべきことはないです。

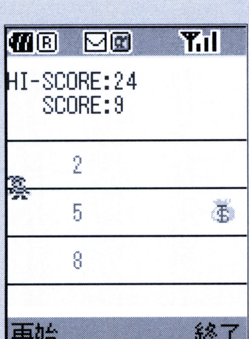
さいごに

ドット絵を描いてくれたM君サンクスです。君が試行錯誤しなければ，警察から逃げるゲームにはならなかったはず(笑)。それから実機での動作確認に協力してくれたnaoさん，Y-Asuka君，どうもありがとうでした。おいらの最高得点は3281です。ホームページ＆メールは以下のとおり。

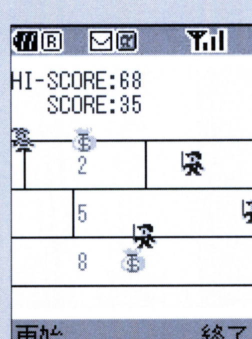
URL http://zura.chu.ath.cx/(iモード http://i.chu.ath.cx/)，メール zurachu@key.homeip.net

★参考にしたもの

・本誌のiアプリ講座&投稿iアプリ
・iモードJavaプログラミング スタンドアロン・アプリケーション編（アスキー)



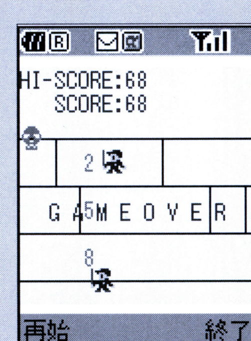
《写真1》画面に表示される3本の線の間を



《写真2》あみだくじの原理で移動していく



《写真3》警官や警察犬にぶつからないように



《写真4》プレイ後のハイスコアは保存される

★（3月号95ページのつぐ美ちゃんとDr.Dのセリフ。僕の場合は）買った！（埼玉県所沢市・**藤16歳**）……【**影**：何を？バレンタインのチョコを自分で？**編**：それは失礼でしょう。影さんじゃあるまいし。**影**：今年は読者の女の子からチョコ届かなかったなぁ。】



《第1図》左から0_gif~4.gifの画像

CHECKER FLAG

つぐ美：あみだくじを利用した移動というのは，以前QBasicの作品でもありましたね。この作品は，自分が作る道以外にもあらかじめ用意された道があるため，進行先が読みにくくて意外と難しく感じるわ。

Dr.D：ステージ表示があみだくじという線のため，わかりにくいかもしれが，あみだ線があるかどうかの判定は，ステージデータをもっている仮想画面配列の値を比較することで行ってお る。細かい点では操作を行うキーを，数字キーや方向キーなど複数に対応させているのに注目。機種によっては方向キーだけだと操作しにくい場合がある。iアプリ制作においては，ぜひとも心がけてほしい処理じゃな。

影：友達に書いてもらったというキャラクタが，単純ながらもいい味をだしているゾ。

《リスト1》amida.javaのリスト

```
//ｱﾐﾀﾞ ｱﾐｰｺﾞ!
//(c)2001 ZR2U
//----------------------------------------------------------------------------
import com.nttdocomo.ui.*;
import com.nttdocomo.util.*;
import com.nttdocomo.io.*;
import javax.microedition.io.*;
import java.util.*;
import java.io.*;

//----------------------------------------------------------------------------
public class amida extends IApplication
{
        public void start()
        {
                myCanvas c=new myCanvas();
                Display.setCurrent(c);
        }
}

//----------------------------------------------------------------------------
class myCanvas extends Canvas implements Runnable
{
        Thread myThread;        //スレッド

        int[][] map=new int[7][20];     //マップ情報
        int[] charaX=new int[2];        //自分と敵のX座標
        int[] charaY=new int[2];        //自分と敵のY座標
        int[] charaVY=new int[2];       //自分と敵のY移動座標
        int iEnemy; //敵の出現フラグ
        int iScore,iHiScore;    //スコア、ハイスコア
        int iGame;  //ゲーム状態フラグ
        int iLevel,iWait=150;   //レベル、ウェイト

        Image[] imgChr=new Image[5];    //画像

        Random rnd=new Random();        //乱数の初期化

//----------------------------------------------------------------------------
        public Image getImg(String sName) //イメージ取得関数
        {
                MediaImage mi=MediaManager.getImage(sName);
                try{
                        mi.use();
                }catch(ConnectionException e){}
                return(mi.getImage());
        }

//----------------------------------------------------------------------------
        public void readSP()    //スクラッチパッド読み出し関数
        {
                try{
                        DataInputStream in=Connector.openDataInputStream("scratchpad:///0");
                        iHiScore=in.readInt();
                        in.close();
                }catch(IOException e){}
        }

//----------------------------------------------------------------------------
        public void writeSP()  //スクラッチパッド書き込み関数
        {
                try{
                    DataOutputStream out=Connector.openDataOutputStream("scratchpad:///0");
                    out.writeInt(iHiScore);
                    out.close();
                }catch(IOException e){}
        }

//----------------------------------------------------------------------------
        public void subInit()  //ゲーム初期化関数
        {
                int i,j;    //ループ用

                for(i=0;i<20;i++){      //マップデータを初期化
                        for(j=0;j<7;j++){
                                map[j][i]=0;
                        }
                }

                charaX[0]=0;            //自分の位置を初期化
                charaY[0]=2;
                charaVY[0]=0;
                iEnemy=0;   //敵出現フラグを初期化
                iScore=0;   //スコアを初期化
                iGame=0;    //ゲーム状態を初期化
        }

//----------------------------------------------------------------------------
        public void subOver()  //ゲームオーバー関数
        {
                writeSP();  //ハイスコアの書き込み

                map[charaY[0]][charaX[0]]+=-map[charaY[0]][charaX[0]]%4+3;          //爆発

                iLevel=0;   //レベルを初期化
                iGame=30;   //ゲームオーバー状態に
        }

//----------------------------------------------------------------------------
        public myCanvas()       //コンストラクタ
        {
                int i;      //ループ用

                for(i=0;i<5;i++){       //画像を読み込む
                        imgChr[i]=getImg("resource:///"+i+".gif");
                }

                readSP();   //ハイスコアの読み出し

                setSoftLabel(Frame.SOFT_KEY_1,"再始");        //ソフトキーのラベルを変更
                setSoftLabel(Frame.SOFT_KEY_2,"終了");

                myThread=new Thread(this);          //スレッド作成・開始
```

iアプリ

入力方法は31ページをご覧ください。

★あの一、PV-7の定価って、29,800円のはずですが（19,800円なのはPV-16のほうです）。（岩手県和賀郡・おねがい＊パニッシュモンスター＊30歳）……【影：恐れ入りました。昔いっしょうけんめいだった世界って忘れないものですよね。私も……なつかしい。】

```
                myThread.start();

                subInit();
        }
//--------------------------------------------------------------------------------
        public void paint(Graphics g)     //ペイントメソッド
        {
                int i,j;    //ループ用

                g.lock();   //画面のロック

                g.setOrigin((getWidth()-120)/2,(getHeight()-120)/2);      //画面サイズの調整
                g.clearRect(0,0,120,120);

                Font f=Font.getDefaultFont();      //フォントサイズの調整
                if(f.getAscent()>13){
                        f=Font.getFont(f.SIZE_SMALL);
                }
                g.setFont(f);

                g.setColor(g.getColorOfName(g.BLACK));          //主線
                g.drawLine(0, 36,119, 36);
                g.drawLine(0, 60,119, 60);
                g.drawLine(0, 84,119, 84);
                g.drawLine(0,108,119,108);

                for(i=0;i<20;i++){      //マップデータに応じて描画
                        for(j=0;j<7;j++)
                        {
                                if(map[j][i]>=4){       //アミダ線
                                        g.drawLine(6*i+6,12*j+24,6*i+6,12*j+48);
                                }

                                if(map[j][i]%4!=0){     //ドル袋・通行禁止・爆発
                                        g.drawImage(imgChr[map[j][i]%4],6*i,12*j+30);
                                }
                        }
                }

                g.setColor(g.getColorOfName(g.RED));            //ボタンのガイド
                g.drawString("2",33, 53);
                g.drawString("5",33, 77);
                g.drawString("8",33,101);

                g.setColor(g.getColorOfName(g.BLUE));           //点数
                g.drawString("HI-SCORE:"+Integer.toString(iHiScore),0,11);
                g.drawString("   SCORE:"+Integer.toString(iScore),0,23);

                if(iEnemy==1){          //敵が存在するなら
                        g.drawImage(imgChr[4],6*charaX[1],12*charaY[1]+30);      //敵
                }

                if(iGame==0){           //ゲーム中なら
                        g.drawImage(imgChr[0],0,12*charaY[0]+30);     //自分
                }
                else{
                        g.setColor(g.getColorOfName(g.BLUE));
                        g.drawString("G A M E  O V E R",15,78);
                }

                g.clearRect(120,0,getWidth()-120,120);          //右にはみ出た部分を消去(D503i用)

                g.unlock(false);        //画面のロック解除
        }
//--------------------------------------------------------------------------------
        public void processEvent(int type,int param) //キープレスイベント
        {
                if(type==Display.KEY_PRESSED_EVENT){
                        switch(param){

                                case Display.KEY_2:     //2or↑が押されている時
                                case Display.KEY_UP:
                                        if((map[1][5]<4)&&(map[3][5]<4)){ //アミダ線
                                                map[1][5]+=4;
                                        }
                                        break;

                                case Display.KEY_5:     //5or選択が押されている時
                                case Display.KEY_SELECT:
                                if((map[1][5]<4)&&(map[3][5]<4)&&(map[5][5]<4)){// アミダ線
                                        map[3][charaX[0]+5]+=4;
                                }
                                break;

                                case Display.KEY_8:     //8or↓が押されている時
                                case Display.KEY_DOWN:
                                        if((map[3][5]<4)&&(map[5][5]<4)){// アミダ線
                                                map[5][5]+=4;
                                        }
                                        break;

                                case Display.KEY_SOFT1:           //ソフトキー1が押されてい
る時
                                        if(iGame==0){
                                                subOver();  //ゲームオーバーへ
                                        }
                                        break;
                                case Display.KEY_SOFT2:           //ソフトキー2が押されてい
る時
                                  IApplication.getCurrentApp().terminate();//iアプリを終了
                                  break;
                        }
                }
        }
//--------------------------------------------------------------------------------
        public void run()      //スレッド実行
        {
                int i,j,k;  //ループ用

                for(;;)     //無限ループ
                {
                        iLevel=iScore/50;     //速度調整
                        if(iLevel>iWait){
                                iLevel=iWait;
                        }
                        try{        //ウェイト
                                Thread.sleep(iWait-iLevel);
                        }catch(InterruptedException ignore){}

                        //ゲーム進行中の場合
                        if(iGame==0)
                        {
                                //自分と敵の移動
                                for(k=0;k<=iEnemy;k++)
                                {
                                        if(charaVY[k]!=0){     //アミダ線を渡る
                                                charaY[k]+=charaVY[k];
                                        }
                                        else{       //主線を渡る
                                                if(k==0){  //自分なら画面スクロール
                                                        for(j=0;j<7;j++){

                                                        for(i=0;i<19;i++){

                                                        map[j][i]=map[j][i+1];
                                                                }
                                                                map[j][19]=0;
```

```
					}

				//新しいマップへのオブジェクトの配置
				j=Math.abs(rnd.nextInt()%30);//アミダ線を張る
					if(j<3){
					map[2*j+1][19]=4;
					}
					//ドル袋or通行禁止を置く
					j=Math.abs(rnd.nextInt()%7);
					i=Math.abs(rnd.nextInt()%20);
					if(i<3){
						map[j][19]+=i;
					}
					//敵が出現していれば敵もスクロール
					if(iEnemy==1){
					charaX[1]--;
					}
					iScore++;  //点数+1
				}
				else{	//敵の場合
					charaX[1]--;
				}
			}

			//自分と敵の当たり判定
			if( (iEnemy == 1) && (charaX[0] == charaX[1]) && (charaY[0] == charaY[1])){
				iEnemy=0;  //敵消去
				subOver(); //ゲームオーバーへ
			}

			if(charaX[k]>=0)
			{
				//マップ物体との当たり判定
				switch(map[charaY[k]][charaX[k]]%4){
					case 1:	//ドル袋
					if(k==0){//自分なら点数+100
					  iScore+=100;
map[charaY[0]][charaX[0]]--;//消去
					}
					break;
					case 2:	//通行禁止
					if(k==0){//自分ならゲームオーバー
					    subOver();
					}
					else{	//敵なら消去
					    iEnemy=0;
					    //爆発を表示
map[charaY[1]][charaX[1]]++;
					}
					break;
				}

				//次の移動先の指定
				if(charaY[k]%2==0){//アミダ線を渡っていない時
					if(charaVY[k]!=0){
					  //アミダ線を渡りきったなら次は主線を渡る
					  charaVY[k]=0;
					}
					else{
					  //下にアミダ線があるなら次は下に移動
					  if(charaY[k]>0){
						if(map[charaY[k]-1][charaX[k]]>=4){
								charaVY[k]=-1;
							}
						}
						//上にアミダ線があるなら次は上に移動
						if(charaY[k]<6){
if(map[charaY[k]+1][charaX[k]]>=4){
								charaVY[k]=1;
							}
						}
					}
				}
				}
			}

			if(iEnemy==0){		//敵の出現
				j=Math.abs(rnd.nextInt()%80);
				if(j<4){
					iEnemy=1;
					charaX[1]=19;
					charaY[1]=2*j;
					charaVY[1]=0;
				}
			}

			if(charaX[1]<0){	//敵が左端に流れたら消去
				iEnemy=0;
			}

			if(iScore>iHiScore){
				iHiScore=iScore;
			}
		}

		//ゲームオーバー時の場合
		else{
			iGame--;
			if(iGame==0){
				subInit();
			}
		}

		repaint(); //再描画
	}
  }
}
```

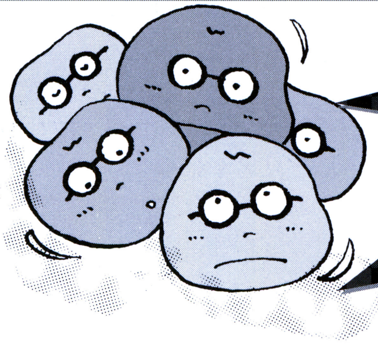
《リスト2》amida.jamのリスト

```
AppName = ｱﾐﾀﾞｱﾐｰｺﾞ!
PackageURL = amida.jar
AppSize = 4283
AppClass = amida
LastModified = Wed, 19 Dec 2001 19:11:46
AppVer = 1.1
KvmVer = CLDC-1.0
SPsize = 16
```

★（3月号56ページ）1人暮らしで、固定電話を持っていない自分には、携帯はメールというより電話機能は必須です。VIVA1人暮らし！（群馬県前橋市・トランス19歳）……【影：最後の文句が気になりますね、つぐ美ちゃん。つぐ美：どういう意味？（ポコスカ）】

iアプリ 入力方法は31ページをご覧ください。

iモード対応携帯電話(iアプリ)用



定番落ち物ゲームのiアプリ版

ぶにょんぶにょん

池添　貴之(19歳)

はじめに

このゲームは見たまんま…の，あの落ち物パズルのパクリです。奇形の物体「ぶにょん」を操作し，同じ色を四つ繋げると消えます。

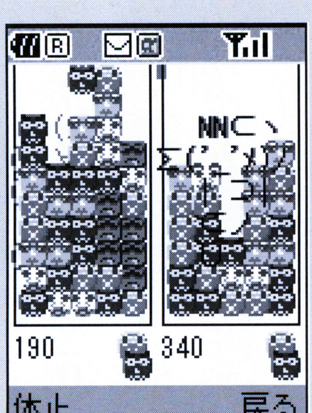
「ぶにょんぶにょん」のオリジナル要素としては，常にアクティブ状態(連鎖や落下中でも操作できる)であったり，特殊な障害オブジェクトがある点です。また，AA(アスキーアート)によるダイナミックなキャラクタのアニメーションもあります。



《写真1》「ぶにょん」を利用した落ち物パズル



《写真2》基本ルールは一般的な落ち物パズルと同じ



《写真3》「障害ぶにょん」などもあり、対戦は白熱必至



《写真4》画面上まで「ぶにょん」が埋まるとゲームオーバー

ゲーム内容

プレイヤーは，上から落ちてくる「ぶにょん」を操作します(**写真2**)。同じ色を四つ以上繋げると「ぶにょん」が消滅し，得点となります。

また，消滅させることで相手に「障害ぶにょん」を送ることができます(**写真3**)。「障害ぶにょん」は3種類あり，ランダムでどれを送るかが変わります。複数の連鎖をさせると，その分送れる量が増えます。

画面が「ぶにょん」で埋まる(正確には真ん中だけ)と負けとなります(**写真4**)。

「おじゃまぶにょん」の説明

おじゃまぶにょん：灰色の「ぶにょん」です。これと隣接して連鎖すると消すことができます。

氷ぶにょん：水色の「ぶにょん」です。これと隣接して連鎖すると，連鎖した「ぶにょん」の色に染まります。

増殖ぶにょん：「おじゃまぶにょん」に酷似。放っておくと，「おじゃまぶにょん」を増殖し，4方向を別の「ぶにょん」で囲むと増殖は止まります。これと隣接して連鎖すると消すことができます。

タイトル画面での設定項目

タイトル画面での設定できるモードなどについての説明です。タイトル画面は，**写真1**のようになります。項目は←→ボタンで選択できます。

モード設定(Player vs CPU，CPU vs

★あっ，プレゼント希望欄に携帯ストラップと書いてしまった。携帯持ってないのに。…こういうことって，当選してから言うべき，か。(福岡県山門郡・**瞬間移動14歳**)……【影：当たった場合は，携帯持っている友達に売りつけるでしょう，やっぱり。】

CPU，Player Only)：Player vs CPUは通常の対戦モードです。CPU vs CPUはその名のとおりCPU同士の対戦を見るだけ…。Player Onlyは，相手がおらず，延々とプレイをするモードです。そのかわり1連鎖ばかりしていると落下速度が上がっていきます（それ以上の連鎖を組めば落下速度は下がります)。

レベル設定：(EASY - AUTO - HARD)：難易度です。HARDになるにつれてCPUが強くなり，AUTOは，こちらの「ぶにょん」の積み具合によって手加減します。また，難易度によって敵のグラフィックが変わります。

Drawing：描画モードを設定します。描画速度が遅い場合は，Full以外を選ぶことをお勧めします。選択できる項目は以下のとおりです。

Full：100%で描画します。エミュレータや速い機種など。

AA Skip：アスキーアートを描画しません。

2P Skip：2Player(CPU)の描画を3/5フレームだけ飛ばします(もちろん描画のみで，処理は飛ばしません。1Pが負けると無効になります)。

Limited：一番軽い描画モードです。アスキーアートと2Ｐ描画(3/5フレーム)を飛ばします。

Vibration(実機でのみ有効)：連鎖時に振動するかを設定します(うっとしいだけですが)。F50x,P50x,SO50x以外の機種ではONにできません。振動している間にBGMが止まることがありますが，これはハードの仕様です。

操作方法

タイトル画面では以下の設定になります。

↑↓ボタン：カーソル移動。
←→ボタン：設定変更。
ソフトキー1，決定ボタン：ゲーム開始。
ソフトキー2：アプリ終了。

ゲーム中では以下の設定になります。

←→ボタン：落ちてくるキャラクタの左右移動。
↓ボタン：落ちてくるキャラクタの高速落下。
↑ボタン，2ボタン：落ちてくるキャラクタの右回転。
決定ボタン，1ボタン：落ちてくるキャラクタの左回転。
ソフトキー1：ゲーム中断(ポーズ機能)。
ソフトキー2：タイトルへ戻る。

苦労した点

容量削減のために泣く泣く関数を削り，個人設定やらの変数群をmyCanvasクラスに入れたため，非常に見えにくいプログラムになってしまいました。その分，コメントやプログラム解説を多く入れたつもりですが…。

あと，iアプリを実機でダウンロードすると，1回でパケット料が20円弱かかるのです…。何かと苦労します(笑)。

詳細

パソコン上ではi-JADEエミュレータ，実機ではP503isにて動作を確認しております。

このプログラムはJBuilder4 Foundationで組んだのですが，かなり使いやすいですよ〜。特に，VisualC++と同じようなキー設定にできるので，VCに慣れた自分にとっては素敵な環境です。さすがに無料のFoundationではjarファイル作成まではできませんでしたが，ソースをきちんとコンパイル，プリベリファイしてjarファイルを作成すれば問題なく実機で動きます。

この「ぶにょんぶにょん」ではBinaryConnectorとRetroGuardを使用して，jarファイルを10KB以内に抑えています。 BinaryConnectorでリソースファイルを_bin.datにまとめることによって，圧縮率を高めることができ，RetroGuardでクラスファイルの容量を数百バイトほど削減することができます。「ぶにょんぶにょん」のビルド方法については，ダウンロードファイル内のbuild.txtに書いておきました。

RetroGuard&BinaryConnectorの使い方や，さまざまな容量削減の方法については，「@ＩＴ」と「ん・ぱか工房」というサイトの記事を参考にしました。

このゲームを作った際に習得したいろいろな容量削減のテクニックをまとめたメモを，うちのHPにアップしておきました。

さいごに

BGMはWindowsのCDに入っていたTown.midをPsmPlayerでMLDに変換したものを使用しています。AAは，「2ちゃんねる」でおなじみの(?)ギコペからコピペし，自分なりにアレンジしたものを使用しています。

インターネットの力はすごいです。プログラマにとってはレベルアップへの最適な手段だといえます。しかし，ネットが使えない人との格差が出てしまうので，そのへんが心配です。これがデジタルディバイドってやつ(ちょっと違うか)？

ベーマガＨＰのプログラム&フリートーク板でアドバイスをくださった皆様に感謝！ホームページやメールアドレスは以下のとおりです。

PsmPlayer http://www.asahi-net.or.jp/~wj3a-fji/phs/program/

Retrologic(RetroGuard) http://www.retrologic.com/

感想，バグ報告i_karisuma@livedoor.com

最新版 http://www.hoops.livedoor.com/~i_karisuma/

★参考にしたもの

・2ちゃんねるのアスキーアート
・IDGジャパン　JAVA World 7月号(使える「iアプリ」の作り方 ― 機種依存の対処)
・ん・ぱか工http://www.people.or.jp/

CHECKER FLAG

影：iアプリながらも超大作扱いの作品。リスト面だけでなく，jarファイル作成にまでこだわった作品だ。
つぐ美：内容はスタンダードな落ち物パズル。だけど，連鎖中にも操作できるなど，このゲーム独自の機能も持っているのが特徴ね。
Dr.D：落ち物パズルに必要な連鎖処理などはもちろん，COM処理までしっかりと作られており，完成度の高い作品といえる。またリスト面では，フォント調整の他，スクラッチパッドを利用した画像作成など，iアプリ作品ならではの工夫も盛り込まれているのが特徴じゃ。

入力方法は31ページをご覧ください。

★なぜか最近のOFコーナー，付録の復活を願うネタが多いですね。ということで，もう一つお願い。付録が復活しても値上げしないでね。（神奈川県川崎市・**桜井道孝21歳**）……【**影**：私を付録に！どう？**つぐ美**：値下げしなきゃ，売れなそう。**影**：ひ，ひどい〜。】

~npaka/
・iアプリJavaゲームプログラミングTips http://www.people.or.jp/~npaka/ibook/tips.html
・@ＩＴ http://www.atmarkit.co.jp/
・DoJaによるiアプリの開発入門 - iアプリを10Kbytesに収めるテクニック(BinaryConnectorはここからダウンロード可能です) http://www.atmarkit.co.jp/fmobile/rensai/doja07/doja07.html

Pick Up LIST

《リスト１》COM思考ルーチン処理部分のリスト

```
// コンピューターの思考ルーチン（行き先param[][8] と回転回数param[][9] を決める）
private static void cpuThink() {
        final byte t[] = { 0,4,2,-2,2,4 };                  // 雑用?!テーブル
        final byte t1 = type[n][0], t2 = type[n][1];        // 操作ぷにょんの色
        int ey[] = new int[7];                                      // 高さ
        int em,max=0,i,j,a,x,y,x2,y2,x3,y3;

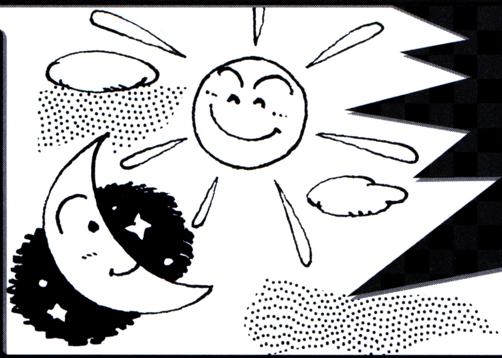
        // まずどこでもいい。
        param[8] = 1+( (gCount%2)<<2 ); param[9] = 0;   // 回転指定なし
        // 色のテーブルを作る
        for( x=6;x>=0;x-- )
        { for( ey[x]=0;m[x][ey[x]+1]==-2;ey[x]++ ); }
        // 上下左右に一番多く接触する位置を探す
        for( x=5;x>0;x— ) { // 右端から調べる
                if( myY[n]+Math.abs(myX[n]-x)<ey[x] ) {
                        i = ( ey[x] + t[x] )/9;
                        for( a=0;a<4;a++ ) {// 回転方向は？
                            y = ey[x]- a%3/2;
                            if( ( a!=3 || x>1 ) && ( a!=1 || x<5 ) && y>0 ) {
                             //初期優先度（EASYになるほど乱数の幅が増える＝精度が落ちる）
                             em = i + Math.abs( rnd.nextInt() )%( 10-( (param[10]*3)>>1 ) );
                             x2 = x+dx[a];
                             y2 = (a%2==0) ? y+dy[a] : ey[x2];
                             for( j=5;j>=0;j-- ) {   // 親ぷにょんと子の周辺をチェック
                                  x3 = x2+dx[j]; y3 = y2+dy[j];
                                  if( j>0 ) {
                                          // 親
                                          if( m[x+dx[j]][y+dy[j]] == t1 ) em += 2-(j>>2);
                                          // 子
                                          if( j<4 ) { if( m[x3][y3] == t2 ) em += 2; }
                                  }
                                  if( j<4 ) {
                                          if( t1==t2 ) {
                                                  // 親と子が同色で隣接する箇所
                                                  if( x3==x && y3==y ) em+=4;
                                          }
                                  }
                             }
                             // 今までで一番em値が高かった
                             if( em>max ) {
                                  max = em; param[8] = x; param[9] = a;
                             }
                             else if( em==max && x==1 ) {
                                  // できれば左端へ…
                                  if( rnd.nextInt()<0 && a==0 ) {
                                          max = em; param[8] = 1; param[9] = 0;
                                  }
                             }
                            }
                        }
                }
        }
        if( gmFlg[2]==1 ) { // Level:AUTO
                // たまにCPUレベルの調整（相手が積まれば油断する。）
                g_param[1-n][10] = ( ey[1]+ey[2]+ey[3]+ey[4]+ey[5] )/10;
        }
}
```

Dr.Dのワンポイント解説

Dr.D：対戦型のプログラムでぜひとも取り入れてほしいのが，このCOM処理。じゃが，どうやって実現したらよいかわからないという読者もいると思うので，ここでCOM処理のリストを取り上げてみた。COM処理も，リストの中身のメインはif文やfor文の組合せで，特別難しい処理をしていないことがわかると思う。つまりCOM処理作成に必要なのは，プログラムの知識でなく，そのCOMの行うアルゴリズムを考えることなのじゃ。

★マンガがわかりやすく，おもしろかった。（福島市・**山岸吉弘33歳**）……【**編**：くり先生，ということです。ますます筆に磨きがかかりそうですね。**くりひろし**：暗に毎月の原稿早くしてと言われてるような。**編**：そんなこと…，少しだけは思ってますが，よろしく！】

ワンダースワン(WonderWitch)用



手軽に遊べるパズルゲーム

月と太陽

畑の木（29歳）

内容と操作方法

ワンダーウィッチで，ちょっとしたパズルを作ってみました。良かったら遊んでみてください。

カラーモノクロ両対応です。カーソルをXボタンで動かし，AもしくはBボタンで，カーソルのある行と列の絵柄を，月なら太陽に，太陽なら月に変えることができます(**写真１**)。画面上のすべての絵柄を，月か太陽どちらかに揃えると，ゲームクリアになります(**写真２**がクリア直前画面)。途中で終了させたくなったら，STARTボタンでタイトルに戻ります。

プログラムについて

内容的には苦労しませんでした。**リスト**を見れば，どこの箇所でなにをしているか(変数名のいい加減さやアルゴリズムの稚拙さを差し引いても)大体わかると思います。

そういえば，４色カラーモードを扱ったのは初めてで，そこでちょっとだけつまづきました。

改造するなら，やっぱり音でしょ。あと，絵柄とか。

パズルについて

このパズルは，有名なものらしくて，例えばゲームボーイ用の『ゼルダの伝説・夢を見る島ＤＸ』のなかで似たようなパズルがあったし，HSP用の『寿司パズル2000』(うすあじさん作)が同じルールだったと記憶しています。だから，私のオリジナルではないことをおことわりしておきます。

ごめんなさい

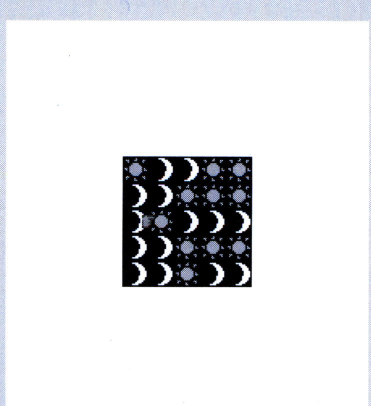
前の掲載作品『(移植版)MIRRORS-FRASH』(本誌2001年10月号掲載)で，ネーム入力のシステムが，WWGP2001グランプリ作品の『ジャッジメントシルバーソード』(M-KAIさん作)を参考…というか真似していたのに，一言もありませんでした。どうも申し訳ありません。

最後に，次回はもっと動きのあるゲームを作りたいなぁと思います。ワンダーウィッチには，スクリーンが二面あるし，スプライトもあるので…(あまり，凝ったものを作ろうとすると，途中で挫折するけど)。

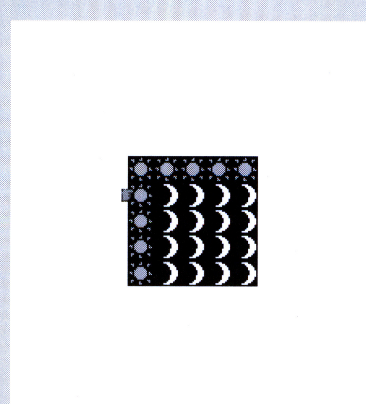
★参考にしたもの

・本誌「チャレンジ！！ワンダーウィッ

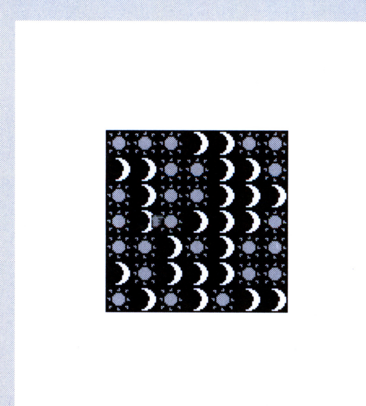
入力方法は33ページをご覧ください。



《写真１》カーソルで指定した列と行のパネルを反転させて



《写真２》すべて同じ絵柄にする。これはクリア直前画面



《写真３》画面サイズは複数用意されている

★高校受験が終わりました。しばらくは遊びまくってます。／本棚がほしい。／ベーマガは良い子にやさしい雑誌ですか？／日めくりカレンダーを買いました。全365ページで満足なボリューム。(東京都練馬区・**どらねく15歳**)……**【影：書きたいだけ書きなさい。】**

チ」のコーナー　電波新聞社

・「ワンダースワンゲームプログラミング」長久勝氏著　ソフトバンク

CHECKER FLAG

つぐ美：チョット久々のWitch作品です。WWGP2002も始まりますし再びWitch熱が加熱するといいですね。本誌でもWitchの作品は随時受付中ですよ～。

Dr.D：パズルの処理としては，仮想画面の数字を反転(この作品では，0か1になる)させるだけなので，map[p] = 1 - map[p];という命令で非常に単純な作りでゲームができる。Witch用ではあるが，C言語によるゲームプログラムの入門用として参考になるリストじゃろう。

編：16×16サイズのBMPファイルをWitch付属ツールで変換して，キャラクタデータの部分を変換したデータに書き換えれば，好きな画像でパズルが楽しめるようになりますよ。

★毎日忙しいです。どうしてでしょう。つぐ美ちゃん，教えて！／私の趣味は無趣味なこと。／はやく2足歩行のペットロボット作って！ソニーさん。（東京都品川区・**大塚まさひろ23歳**）……**【つぐ美：最初の問いかけにお応えします。それは忙しくしてるから。】**

《リスト1》M&S.cのリスト

```
/*****************************************************************************
        簡易パズル「月と太陽」for Wonder Witch
                                            by 畑の木
*****************************************************************************/
#include <stdlib.h>
#include <sys/bios.h>
#include <sys/libwwc.h>

#define CHN_SP     0                        /* キャラクタ番号:スペース */
#define CHX_SP     1                        /* キャラクタ数:スペース */
#define CHN_CR     (CHN_SP + CHX_SP) /* キャラクタ番号:カーソル */
#define CHX_CR     1                        /* キャラクタ数:カーソル */
#define CHN_SN     (CHN_CR + CHX_CR) /* キャラクタ番号:太陽 */
#define CHX_SN     4                        /* キャラクタ数:太陽 */
#define CHN_MN     (CHN_SN + CHX_SN) /* キャラクタ番号:月 */
#define CHX_MN     4                        /* キャラクタ数:月 */
#define CHR_W      2                        /* キャラクタの横幅:月と太陽 */
#define CHR_H      2                        /* キャラクタの縦幅:月と太陽 */

#define SPN_CR     0                        /* スプライト番号:カーソル */
#define SPN_MAX    1                        /* スプライト番号:最大使用数 */

#define CPN_CR     12        /* カーソル表示用のパレット */
#define CPN_MS     13        /* 月と太陽用のパレット */

#define CPN_SP     8         /* キャラクタパレット番号:スプライト有効開始値 */
#define CFM_CPN    ((CPN_CR - CPN_SP) << CFSFT_PALETTE) /* パレット番号割り振り */

#define MAP_MIN    5         /* ステージ上に置かれるキャラの最小使用数 */
#define MAP_MAX    7         /* ステージ上に置かれるキャラの最大使用数 */
#define END_FLAG   3

/* 乱数初期化 */
#define randomize (srand(rtc_get_datetime(RTC_SEC) + (rtc_get_datetime(RTC_MIN) <<
8)))
#define random(x)  (rand() % (x))      /* 0 から (x - 1) までの乱数発生 */
#define cls screen_fill_char(SCREEN2, 0, 0, 32, 32, 0)  /* 画面クリア */
#define erase_sprite(sp_n) sprite_set_location(sp_n, 28 * 8, 18 * 8) /* スプライト消去 */

short sunBK[CHX_SN], moonBK[CHX_MN];
short map[MAP_MAX * MAP_MAX];
int game_map_max;

unsigned short space_chr[] = {        /* 空白のデータ */
        0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
};
unsigned short sign_chr[] = {         /* カーソルのデータ */
        0x5E20, 0x817E, 0x0EF0, 0x8478, 0x8C70, 0x8478, 0x8870, 0x7000,
};
unsigned short sun_chr[] = {          /* 太陽のデータ */
        0xFFFF, 0xFFFF, 0xCEFF, 0xEFFF, 0xFCFF, 0xF8FF, 0xF0FF, 0x90FF,
        0xFFFF, 0x7FFF, 0x73FF, 0xF7FF, 0x3FFF, 0x1FFF, 0x0FFF, 0x0BFF,
        0xD0FF, 0xF0FF, 0xF8FF, 0xFCFF, 0xCFFF, 0xDEFF, 0xFEFF, 0xFFFF,
        0x09FF, 0x0FFF, 0x1FFF, 0x3FFF, 0xF3FF, 0xFBFF, 0x7FFF, 0xFFFF,
};
unsigned short moon_chr[] = {         /* 月のデータ */
        0xFFFF, 0xFFFC, 0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
        0xFFFF, 0xFF3F, 0xFF0F, 0xFF87, 0xFFC3, 0xFFC3, 0xFFE1, 0xFFE1,
        0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF, 0xFFFC, 0xFFFF,
        0xFFE1, 0xFFC1, 0xFFC3, 0xFFC3, 0xFF87, 0xFF0F, 0xFF3F, 0xFFFF,
};

int Pdata1[] = {  /* ワンダースワンカラー用のパレット（カーソル用） */
        0x000, 0xAAA, 0x666, 0x000
};

int Pdata2[] = {  /* ワンダースワンカラー用のパレット（月と太陽用） */
        0x000, 0xf00, 0xff0, 0x000
};

/*****************************************************************************
        初期設定
*****************************************************************************/
void init()
{
        int i;

        randomize;

        if (wwc_get_hardarch() == HARDARCH_WSC) {
                wwc_set_color_mode(COLOR_MODE_4COLOR);
                for(i = 0; i < 4; i++) {
                        wwc_palette_set_color(CPN_CR, i, Pdata1[i]);
                        wwc_palette_set_color(CPN_MS, i, Pdata2[i]);
                }
        } else {
                palette_set_color(CPN_CR, 0x7710);
                palette_set_color(CPN_MS, 0x5200);
        }
```

```
	display_control(DCM_SPR | DCM_SCR2);

	sprite_set_range(SPN_CR, SPN_MAX);
	font_set_colordata(CHN_SP, CHX_SP, space_chr);

	font_set_colordata(CHN_CR, CHX_CR, sign_chr);
	sprite_set_char(SPN_CR, CHN_CR | CFM_CPN | CFM_SPR_UPPER);

	for(i = 0; i < 4; i++) {
		sunBK[i] = (CPN_MS << CFSFT_PALETTE) | CHN_SN + i;
		moonBK[i] = (CPN_MS << CFSFT_PALETTE) | CHN_MN + i;
	}
}

/**************************************************************************
	タイトル表示、モードの選択
**************************************************************************/
int title()
{
	int m = 0, k;

	text_screen_init();

	text_put_string(13, 5, "月と太陽");
	text_put_string(11, 8, "5*5 STAGE");
	text_put_string(11, 8, "SQUARE(5)");
	text_put_string(11, 9, "SQUARE(6)");
	text_put_string(11, 10, "SQUARE(7)");
	text_put_string(11, 11, "EXIT");

	do {
		sys_wait(1);
		k = key_hit_check();
		m += ((m < 3) * (k == KEY_X3) - (m > 0) * (k == KEY_X1));
		sprite_set_location(SPN_CR, 9 * 8, (8 + m) * 8);
	} while (!(k & KEY_A) && !(k & KEY_B) && !(k & KEY_START));

	if (k & KEY_START) m = END_FLAG;
	return m;
}

/**************************************************************************
	仮想画面の数値（図形をあらわす）を変える
**************************************************************************/
void change(int p)
{
	map[p] = 1 - map[p];
}

/**************************************************************************
	仮想画面上で、指定した行と列の図形を変える
**************************************************************************/
void change_map(int x, int y)
{
	int i;

	change(x + game_map_max * y);
	for(i = 0; i < game_map_max; i++) {
		change(x + game_map_max * i);
		change(y * game_map_max + i);
	}
}

/**************************************************************************
	図形がそろったか、判定
**************************************************************************/
int hantei()
{
	int a;
	int i;

	for(i = 0, a = 0; i < game_map_max * game_map_max; i++)
		a += map[i];

	if (a == 0 || a == game_map_max * game_map_max)
		return 1;
	else
		return 0;
}

/**************************************************************************
	ゲームスタート時の図形のシャッフル
**************************************************************************/
void shuffle()
{
	int a, x, y;
	int i;

	for(i = 0, a = random(2); i < game_map_max * game_map_max; i++)
		map[i] = a;

	for(i = 0; (i < 100 || hantei()); i++) {
		x = random(game_map_max);
		y = random(game_map_max);
		change_map(x, y);
	}
}

/**************************************************************************
	ゲーム開始時の初期
**************************************************************************/
void game_init()
{
	shuffle();

	key_set_repeat(8, 20);

	cls;
```

★友達にすすめられて，はじめて買いました。大学入試にBASICの問題が出てるんですね。何人が選択したんでしょうか。（埼玉県川越市・とも男15歳）……【編：選択した人，レポートくださいね。待ってます。Dr.D：来年大学受験の読者諸君は，今から特訓だ！】

入力方法は33ページをご覧ください。

★OFって、全ページ掲載はできないのですか？載るとみんな買うと思いますよ、って余計なお世話ですね。(新潟県新発田市・こうぞう18歳)……【編集長：そこまで気を使ってくれて、ありがとう。というわけだ、影くん。影：ど、どうして私にふるんですか。】

```
	font_set_colordata(CHN_SN, CHX_SN, sun_chr);
	font_set_colordata(CHN_MN, CHX_MN, moon_chr);
}

/*****************************************************************************
	図形の画面表示処理
*****************************************************************************/
void set_map(int m_x, int m_y)
{
	int i;
	int x, y;

	for(i = 0; i < game_map_max * game_map_max; i++) {
		x = m_x + (i % game_map_max) * 2;
		y = m_y + (i / game_map_max) * 2;
		if (map[i] == 0)
			screen_set_char(SCREEN2, x, y, CHR_W, CHR_H, sunBK);
		else
			screen_set_char(SCREEN2, x, y, CHR_W, CHR_H, moonBK);
	}
}

/*****************************************************************************

*****************************************************************************/
void clear(int m_x, int m_y)
{
	int i, j;

	erase_sprite(SPN_CR);
	sys_wait(10);

	for(i = 0; i < 10; i++) {
		for(j = 0; j < game_map_max * game_map_max; j++)
			change(j);
		set_map(m_x, m_y);
		sys_wait(10);
	}

	text_screen_init();
	text_put_string(11, 9, "クリア !!");
	key_wait();
}

/*****************************************************************************
	ゲームのメイン処理
*****************************************************************************/
void game(int mode)
{
	int k;
	int cur_x = 0, cur_y = 0;
	int m_x, m_y;

	game_map_max = MAP_MIN + mode;
	m_x = 14 - game_map_max;
	m_y = 9 - game_map_max;

	game_init();
	set_map(m_x, m_y);

	do {	/* カーソルの移動 */
	    k = key_hit_check_with_repeat();
	    if ((cur_x < game_map_max - 1) && (k & KEY_X2)) cur_x++;
	    if ((cur_x > 0)                && (k & KEY_X4)) cur_x--;
	    if ((cur_y < game_map_max - 1) && (k & KEY_X3)) cur_y++;
	    if ((cur_y > 0)                && (k & KEY_X1)) cur_y--;
	    sprite_set_location(SPN_CR, m_x * 8 + cur_x * 16 - 4, m_y * 8 + cur_y * 16 + 4);

	    if ((k & KEY_A) || (k & KEY_B)) {		/* 図形を入れ替える */
		change_map(cur_x, cur_y);
		set_map(m_x, m_y);
		if (hantei()) {		/* 判定 */
			clear(m_x, m_y);
			break;
		}
	    }
		sys_wait(1);
		if (k == KEY_START) break;  /* ギブアップkey */
	} while (1);
}

/*****************************************************************************
	メインルーチン
*****************************************************************************/
main()
{
	int mode;

	init();

	for(;;) {
		if ((mode = title()) == END_FLAG) return;
		game(mode);
	}
}
```

《リスト2》M&S.cfのリスト

```
name: moon&sun
info: 月と太陽
mode: 7
source: M&S.bin
output: M&S.fx
```

★★★★★★★ マイコンBASICマガジン ★★★★★★★

明日のスター・プログラマー

Programmer's Room

今月のベスト・プログラマー

編：新学期号にふさわしく，出来の良い投稿作品が集まった印象です。いい意味でベストの選考に悩みますね。

Dr.D：うむ，特に今月の作品で注目してほしいのが，VB作品のフォーム部分じゃ。ほとんどの作品が，フォームをコンパクトに作成してある。やはりベーマガでは，こういった入力面に関する部分にも気を配ってほしいからのぉ。これはなかなかうれしい傾向じゃな。

つぐ美：作品の完成度に注目すると，やはり3Dシューティングの『3D Shoot!』や，iアプリの『ぷにょんぷにょん』などが目につきますね。これぞ超大作作品って感じです。

編：う～ん，やはり意見が分かれますね。そんななか選び抜いたベスト・プログラムはどの作品でしょう，Dr.？

Dr.D：今月はVisualC++用の『dust』をベストに選んでみた。内容もさることながら，やはりリスト面で目についたこの作品が，ベーマガでのベストにふさわしいのではないかな？

つぐ美：というわけで今月のベスト・プログラマーは，『dust』の作者，kouさんに決定です。おめでとうございます。

影：Dr.も認めた『dust』のプログラムリストは，「Dr.Dのベストプログラム研究室」でバッチリ解説するゾ。

投稿プログラムコーナーの説明

編：４月号ということで，私たちも初心に戻って，この投稿プログラムコーナーや投稿方法について，今一度説明していこうと思います。

つぐ美：この投稿プログラムコーナーというのは，読者のみなさんが作成したプログラム作品の投稿を受け付け，毎月の投稿のなかから，優秀な作品を紹介していくというコーナーです。

影：Web上などのソフトを紹介するコーナーは，他の雑誌にもあるけど，投稿作品を，リスト付きで掲載していくってのは，もはやベーマガくらい。プログラム好きな人には，非常にありがた～いコーナーなのです。

編：しかも採用された際には，誌面への掲載はもちろん，Dr.Dをはじめ私達がリスト面や作品の内容についてCHECKER FLAGでポイントを評価しています。さらに，あまり知られていないのかもしれませんが，何と作品が掲載された投稿者には，掲載誌と原稿料をお送りしています。

つぐ美：この原稿料は，何と１万円から税金を引いた金額＝９千円が送られてくるんですよ～。けっこう大きいでしょ？この金額を見て，原稿料をGetしたいと思ったあなた。ぜひプログラム投稿に参加してくださいね。

編：それでは，その作品の投稿方法について簡単に説明していきましょう。まず，何回も言っていますが，投稿に関する規定はプログラムコーナーの扉（29ページ）に書いてありますので，じっくり読んでから投稿してきてください。郵送の他，メールでも受け付けています。メールでの詳しい説明は，ベーマガＨＰ上に記載されています。

影：投稿する前には，過去に掲載された作品をしっかり研究することも大切だ。作品だけでなく，説明文の項目や掲載されているリストや画像などもしっかりチェックして，不備のない投稿をするように心がけることが大切だゾ。

編：あと，投稿の際には，投稿カードのウラ面や，説明ファイルの最後の方などに，投稿コーナーに関するご意見があると非常にうれしいです。

Dr.D：心機一転のこの時期。皆からの新鮮味あふれる投稿作品を，編集部一同心待ちにしておるゾ。

今月はもう一歩でした

●HSP用『Hit&Blow』／中島武志 (27歳)

つぐ美：ウィンドウのタイトルバーのみを用いて実行するというソフト。実行形態はユニークでしたが，内容はもう一歩。正解数だけでなく，入力タイムなども表示されるといいですね。あと，投稿は，ソースファイルを添付した形式でお願いいたします。

●VB+DirectX用『ミーちゃんのハンマー』／橘　一馬 (29歳)

影：プレイする前に，ゲームクリア表示が出るバグがあった。おそらくタイマー関連処理のミスじゃないかな？タイマーを使ったプログラムの場合は，ある程度遅い機種でも動作するよう考慮しよう。 あと，もう少し説明文を充実させてくれるとありがたい。

● VisualC++用『Backup Manager』／福崎忠彦(26歳)

Dr.D：非常に出来の良い作品なのじゃが，基本的にバックアップを行うことがメインのソフトなので，実用性という面が疑問が残ってしまった。

● VisualC++用『予測能トレーナー』／お名前の記入がありませんでした

編：反射神経を使った遊びやすい作品でしたが，この作品の特徴であるマス目の変化がチョットわかりにくいのが難点。また，お名前の記入は忘れずに！

投稿ありがとう

岩崎秀彦／ら一抹茶／内田　寛／kou／田村良一／畑の木／ヅラChu／橘　一馬／伊藤　優／中島武志／福崎忠彦／山内喜智／石山明弘／片桐貴志／ロディマス／三好利治／池添貴之／フリーランス／松本輝彦／
※全員紹介できなくてこめんなさい

■で・ば・ぐ　2002年3月号82ページ掲載「熱帯魚」のプログラムにバグがありました。50行の"　　　"内のスペースは不要です。ごめんなさい。

Dr.Dのベスト・プログラム研究室



今月のベスト・プログラム：VisualC++用『dust』（49ページ掲載）

今月のベスト・プログラムは，VisualC++用『dust』。破片による連鎖破壊を行うゲーム性はもちろん，丁寧に書かれたリストも評価できる作品じゃ。今回は，このゲームの特徴である破片の移動処理と，さらに，改造関連の解説を強化して，説明文で触れている背景処理と最大連鎖数の表示，そして弾の連射を可能にする処理の追加と，計三つの改造を紹介していこう。

(Dr.D)

破片の移動処理

破片の移動処理というのは，爆発時の移動を示す。この作品では，破片は放物線を描きながら移動するというキレイな形に仕上がっている。ここでは，この破片の移動処理を解説していこう。

破片移動処理の前に，まずは，この作品内での移動処理について解説しよう。今回のプログラムでは，XY方向それぞれへの移動量を示す変数を定義しておき（構造体の中で定義されている），その値を画面幅に合わせて加算もしくは減算を行うという形で移動処理を行っている。

破片の移動も，常に画面下へ移動させる処理が加わるが，基本的な処理はこれを同じ方法で実現しておるのじゃ。では，なぜ加算減算だけで放物線が描けるのか説明しよう。このプログラムでは，出現する四つの破片の移動量を**第１表**のように設定しているのじゃ。ポイントはY方向の移動量にマイナスの値を設定している点じゃな。マイナスの値を設定することで，一定時間は画面上へ破片が移動することになり，結果として等速運動の放物線が描かれているというわけじゃ。

《第１表》移動量の設定値

破片の位	X方向	Y方向
左上	-2	-3
左下	-2	0
右上	2	-3
右下	2	0

改造してみよう

背景処理の追加

今回作成する背景は，説明文でも例に挙げている星のスクロールを表示するものにしてみた。この背景処理の実現方法は，星のスクロールを行う関数を新たに定義して，画面表示の時にその関数を呼び出すようにすればよいのじゃ。その星のスクロールを行う関数の記述例を**リスト１**に示してみた。この関数の細かな説明は割愛するが，要は定義された数だけ，星に見立てたドットを画面下へ移動させながら描画させるのじゃ。

最大連鎖数の表示

この改造は，基本的には，スコア加算と一緒の時に連鎖数を示す変数の値を＋１していけばよいのじゃが，一つ注意しなければならない点がある。それは，連鎖が止まった時（画面から破片がなくなった時），それまでの連鎖数を最大連鎖数として別途保存する必要があるという点じゃ。この処理を入れないと，ゲームが終了するまでの連鎖数をカウントしてしまうことになるのじゃ。

それでは，この点を踏まえてコーデ

《リスト1》星のスクロールを行う関数のリスト

```
//星の最大数を示す  #define Stars 50  を定義しておく
//この関数の使用方法は，Syokika関数内で初期化用にstarscroll(0)
//Draw関数内で，表示用にstarscroll(1)と記述する
void starscroll(int n)
{
		static int f[Stars],x[Stars],y[Stars]; //星の状態を示すフラグと座標
		long color; //色情報
		int i,m,k; //雑用

		//スタート時の座標設定
		if(n==0){
				for(i=0;i<Stars;i++){
						f[i] = (rand() % 5)+1; //星の出現＆移動スピードを示す
						x[i] = (rand() % 380); y[i] = (rand() % 15)-15; //座標設定
				}
		}
		//スクロール処理
		else{
				for(i=0;i<Stars;i++){
						if(f[i]==-1){ //星が存在しない時は，上と同じ処理で初期化
								f[i] = (rand() % 5)+1;
								x[i] = (rand() % 380); y[i] = (rand() % 15)-15;
						}
						//色設定(白：赤＝３：１にしている)
						if((i % 4)==0)
								color = RGB(255,0,0);
						else
								color = RGB(255,255,255);
						//移動 f[i]の値だけ移動
						y[i] = y[i]+f[i];
						//行列設定＆描画
						for(m=0; m<2; m++)
								for(k=0; k<2; k++)
										SetPixel( backdc, x[i]+m, y[i]+k, color );
						//画面外まで行ったら初期化
						if(y[i]>385)
								f[i]=-1;
				}//for
		}//else
}
```

ィング面での説明を行っていこう。まず，基本はスコアと同じ記述なので，スコア処理と同じように，グローバル変数の定義と値の初期化，ウィンドウ部分への値表示，連鎖数の加算処理などを記述しよう。そして，説明したように連鎖が途切れた時の最大連鎖数の処理を記述する。これは，破片の状態を示すフラグを調べ，すべて待機状態だったら連鎖が途切れているということなので，今までの連鎖数が最大であるかどうかの確認処理を行う，という流れになる。この部分も，必要な部分をまとめて**リスト2**に掲載しているので，合わせて参照してほしい。

弾の連射処理

最後は，説明文の改造の項目では触れてはいないが，弾の連射を実現させる改造についてじゃ。連鎖を実現させるには，弾の座標などを配列で定義しなければならない。じゃが，このプログラムでは弾のデータと破片のデータを同じ配列で定義して処理を行っている。この配列の添え字が0の時は弾である，という形式で組まれているのじゃ。ということは，弾を示す添え字の範囲を増やせば，その分だけ弾のデータが増える＝連射ができるというわけじゃ。この考えに基づいて記述した連射処理を**リスト3**に掲載してみた。添え字の範囲拡大に合わせて，配列データを扱うループの範囲を変更すれば問題なく実現できるはずじゃ。

今回は，改造項目の解説が多くなってしまい，リスト面での解説が若干少なくなってしまった。じゃが，このプログラムは，非常にキレイな形式で書かれておるため，C言語をある程度理解している読者ならリストのコメントから処理内容が理解できると思う。C言語でのゲームプログラムに挑戦している読者は，ぜひとも1度は目を通してほしいリストじゃ。

また，三つの改造を紹介したが，このゲームは，まだまだいろんな改造ができるプログラムじゃ。ぜひとも改造にチャレンジしてほしい。素晴らしい改造プログラムができたなら，ぜひとも編集部宛にも送ってきてほしい。内容によっては，投稿プログラムコーナーでも紹介していきたいと思っておる。



▶《写真1》三つの改造を行ったプログラムの実行画面

《リスト2》最大連鎖数処理を行う部分のリスト

```
//グローバル変数で定義
long maxnum = 0, high_maxnum = 0;

――――――――――――――――――――――――――――

//Draw関数内に記述する処理　連鎖数をウィンドウバーに表示
sprintf(text, "high_score:%d score:%d max:%d high_maxnum:%d",
		high_score, score, maxnum, high_maxnum);
SetWindowText(hwnd, text);

――――――――――――――――――――――――――――

//WinMain関数内に記述する処理　最大連鎖数処理
if (game_state == 0) {//タイトル
		if (key[13])//エンターキーでスタート
				game_state = 1;
		score = 0; maxnum = 0;//初期化処理
}
else if (game_state == 1) {//ゲーム中
		a_time = Appear(a_time);// 敵出現
		keyin();//ゲーム中のキー処理
		Move();//移動処理＆当たり判定

		//以下の処理で，ゲーム中に連鎖が途切れた時の連鎖最大値保存を行う
		for (i = 0; i <= Dnum; i++){ //すべてのフラグ状態を調べる
				if (dust[i].Flag == 0) j++;
				else break;
		}
		if(j == Dnum + 1){//画面上に弾と破片がない状態(連鎖が途切れた)
				if (high_maxnum < maxnum) high_maxnum = maxnum; //最大数であるかの判定
				maxnum = 0; //現在の連鎖数の初期化
		}
		j = 0; //次の処理のためjを初期化

//途中省略

if (j == Dnum + 1) { //ゲームオーバー時のハイスコア判定
		if (high_score < score) high_score = score;
		if (high_maxnum < maxnum) high_maxnum = maxnum;

――――――――――――――――――――――――――――

//Move()関数内に記述する処理　連鎖数の加算
//スコア加算処理のある，Burst関数と同じ場所に記述

//敵との当たり判定
if (chara[0].Flag != 0 && i != 0)
		if (chara[0].x-16 <= chara[i].x && chara[i].x <= chara[0].x+16
			&& chara[0].y-16 <= chara[i].y && chara[i].y <= chara[0].y+16){
				Burst(0, (int)chara[0].x, (int)chara[0].y, dust[i].value);
				maxnum++;　//値を＋1
		}
}
```

《リスト3》連射処理関連のリスト

```
//連射数を定義
#define Shot_MAX 5
#define Dnum Shot_MAX + 120 //弾＆破片の数

――――――――――――――――――――――――――――

//キー入力部分
void keyin()
{
		static keywait=0; //連射間隔調整用のstatic変数
		int i; //連射処理を行うループ用変数

		//途中省略

		if(keywait>0) //連射間隔調整
				keywait--;
		//弾を発射する
		if (chara[0].Flag != 0 && key[32] && keywait==0) {
				for(i=0; i<Shot_MAX; i++){ //弾の数だけ処理を行う
						if (dust[i].Flag == 0) {
								dust[i].Flag = 1;
								dust[i].x = chara[0].x;
								dust[i].y = chara[0].y - 16;
								dust[i].xstep = 0;
								dust[i].ystep = -10;
								dust[i].pict = 0;
								dust[i].value = 1;
								keywait=10;
								break;
						}
				}
		}
}

――――――――――――――――――――――――――――

//ループ関連の修正部分(Move関数)

//i != 0 を以下のように修正
if (i >= Shot_MAX)　//破片は下に向けて加速
		dust[i].ystep += 0.1;

――――――――――――――――――――――――――――

//ループ関連の修正部分2(Burst関数)

//変数jの初期化を以下のように修正
int j = Shot_MAX; // 1をShot_MAXに変更
```

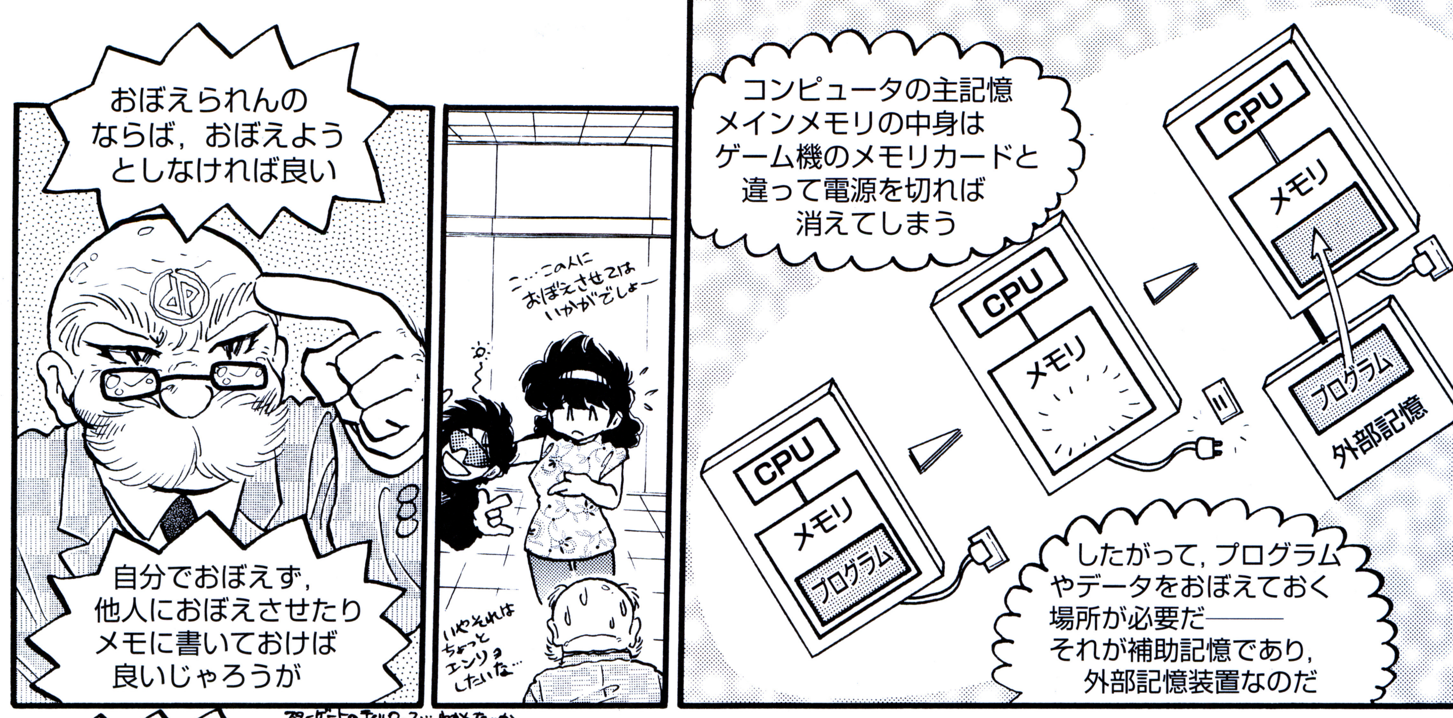
マンガで学ぶ コンピュータの世界

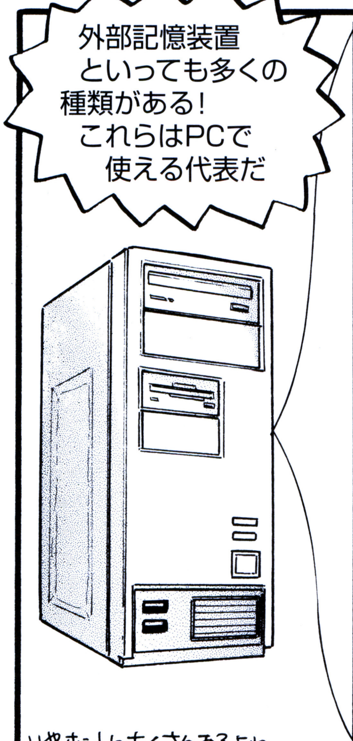
パソコンレクチャー ザ・ビギニング

「ハードディスクは忘れない!!!」

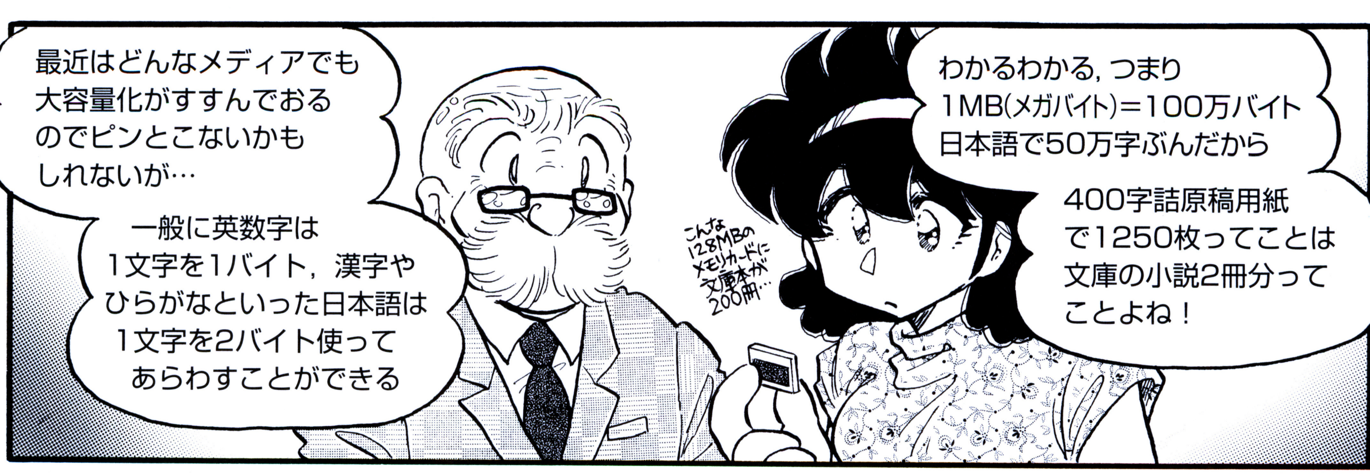
by くりひろし

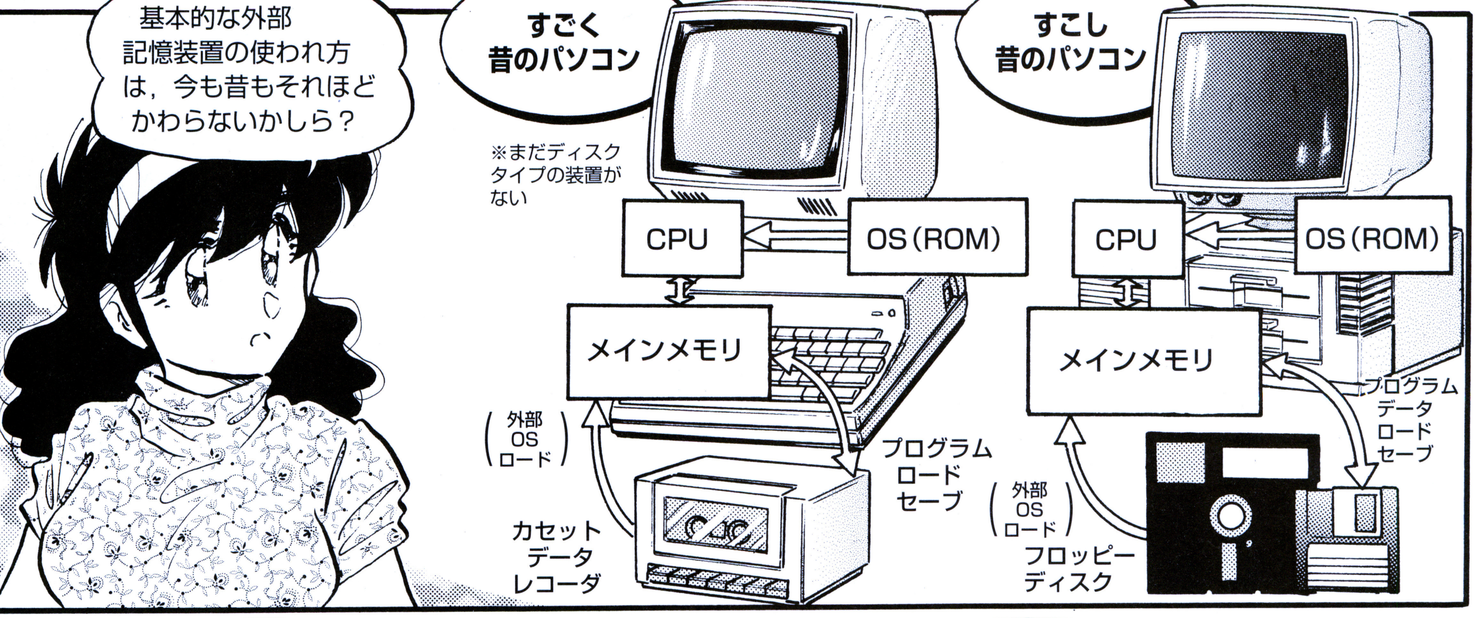




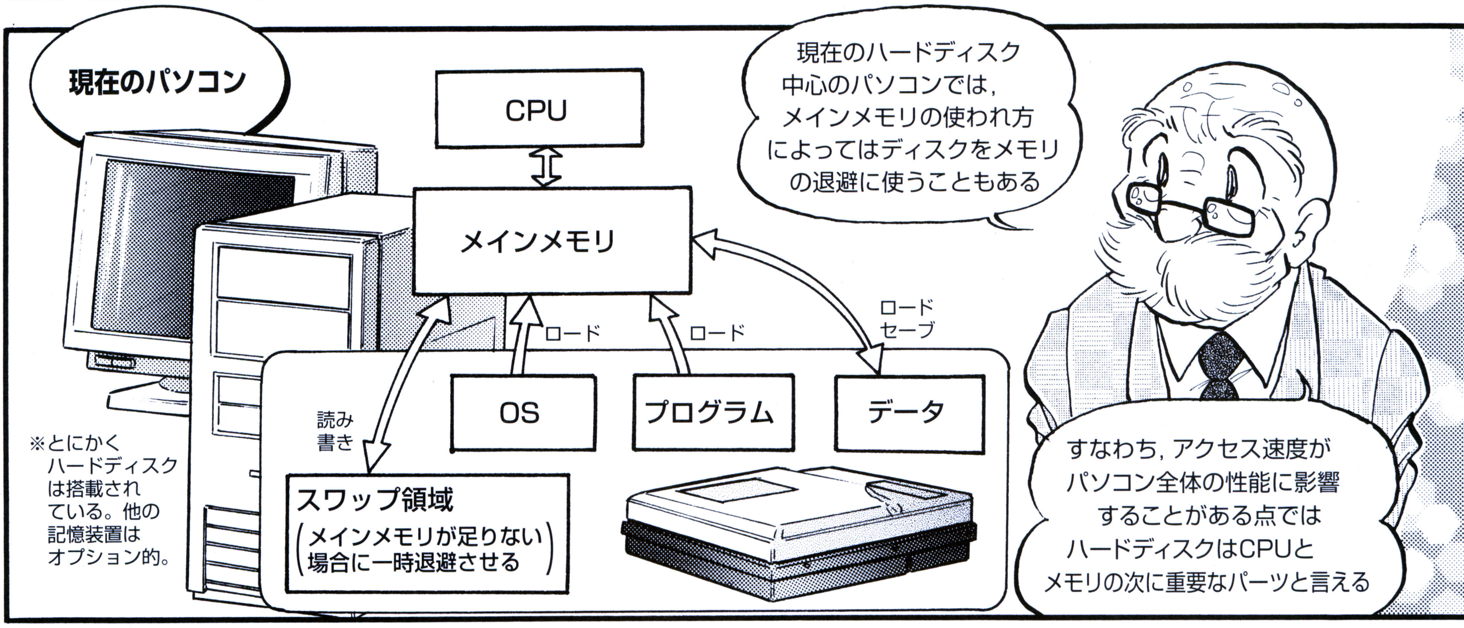


	規　格	容　量	特　徴
磁気ディスク・フロッピーディスク	3.5インチスーパーディスク 3.5インチフロッピー 5インチフロッピー 8インチフロッピー	120MB 1.44MB（HD） 1.2MB（2HD） 640KB	OSがCD-ROMから起動できる現在，フロッピーレスのノートブックPCも多く今や過去の資産になりつつある。
磁気ディスク・ハードディスク	2.5インチ 3.5インチ 5インチ マイクロドライブ	～60GB ～160GB ～40MB 1～2GB	主流は7200回転の高速型かつ静音性にもすぐれた流体軸受けモータを採用したもの。大容量化はそろそろ頭打ちでこれからは小型化に流れる。
光ディスク／光磁気ディスク	CD-ROM/R/RW DVD-ROM/R/RAM/RW MO ZIP	540～800MB 4.7～9.4GB ～1.3GB 100MB	CD-RWは手軽だがデータの記録用はDVD系に。ただし規格が乱立していて互換性に要注意。使いやすさと保存性ではMOもまだまだ使える。
メモリデバイス	PCカード/CF SD/MMC/MS USBメモリ	～1GB ～256MB ～256MB	手軽でデジカメやMP3プレーヤとのデータのやりとりにも便利。ダークホースがUSBメモリか？
磁気テープ	DAT 8ミリテープ	1～2GB	ワークステーション等で使われる事があるがPCでは一般的でない

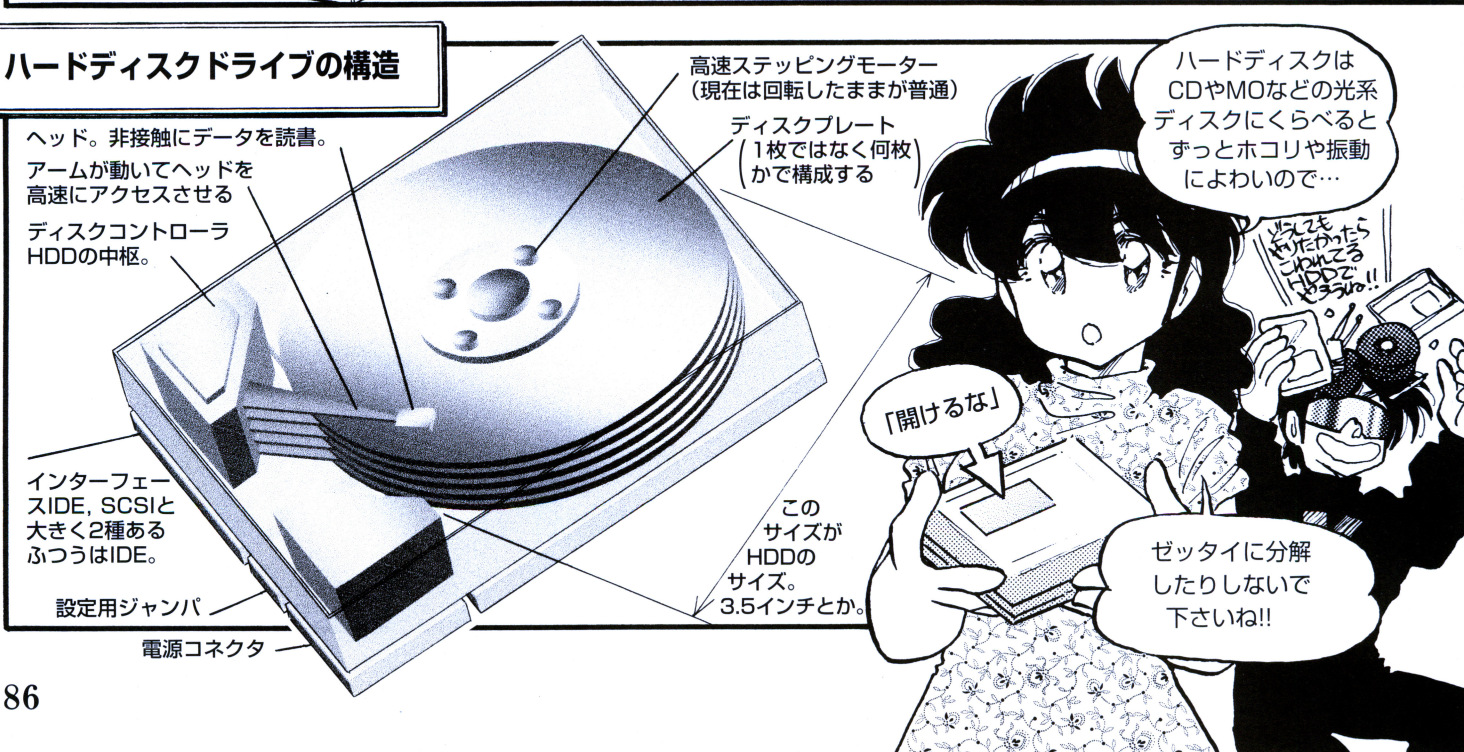




基本的な外部
記憶装置の使われ方
は，今も昔もそれほど
かわらないかしら？
すごく
昔のパソコン
※まだディスク
タイプの装置が
ない
CPU
OS（ROM）
メインメモリ
外部
OS
ロード
プログラム
ロード
セーブ
カセット
データ
レコーダ
すこし
昔のパソコン
CPU
OS（ROM）
メインメモリ
プログラム
データ
ロード
セーブ
外部
OS
ロード
フロッピー
ディスク



現在のパソコン
CPU
メインメモリ
ロード
ロード
ロード
セーブ
読み
書き
OS
プログラム
データ
スワップ領域
（メインメモリが足りない
場合に一時退避させる）
※とにかく
ハードディスク
は搭載され
ている。他の
記憶装置は
オプション的。
現在のハードディスク
中心のパソコンでは，
メインメモリの使われ方
によってはディスクをメモリ
の退避に使うこともある
すなわち，アクセス速度が
パソコン全体の性能に影響
することがある点では
ハードディスクはCPUと
メモリの次に重要なパーツと言える



ハードディスクドライブの構造
高速ステッピングモーター
（現在は回転したままが普通）
ディスクプレート
（1枚ではなく何枚
かで構成する）
ヘッド。非接触にデータを読書。
アームが動いてヘッドを
高速にアクセスさせる
ディスクコントローラ
HDDの中枢。
インターフェー
スIDE，SCSIと
大きく2種ある
ふつうはIDE。
設定用ジャンパ
電源コネクタ
この
サイズが
HDDの
サイズ。
3.5インチとか。
ハードディスクは
CDやMOなどの光系
ディスクにくらべると
ずっとホコリや振動
によわいので…
どうしてもやりたかったらこわれてるHDDでやろうね!!
「開けるな」
ゼッタイに分解
したりしないで
下さいね!!



市販最大容量
1MB当たり
200GB
100GB
40円
30円
20円
10円
1990 1992 1994 1996 1998 2000 2002
5インチ
IDE
SCSI
3.5インチ
E−IDE
W−SCSI
2.5インチ
静音化
マイクロサイズ化
次世代化
1990年頃だと1MBあたり1万円以上していた
200〜250GBより上は新しい技術が必要そうだ
パソコンでビデオ録画やAV編集があたり前になり，ついに100G超に
現在最大160GB
OSもソフトも大容量化し，急激にHDDも大容量化していく
WIN95後のパソコンブームでついに1MBあたりで10円を突破した
今や1MBあたり1円以下…
今どきのHDDはパソコンに不可欠なだけでなく，長時間のビデオ録画にも使えるほど最も大容量のメディアだわね
こんなに小さくても5GB♡
大容量と高速性が生かされて，PCなしで単体のビデオレコーダにも使われるようになってるのはちょっとしたオドロキよねーー
★RAIDディスクとは？
PC
RAIDコントローラ
#0
#1
同じ内容
同じ内容
これはRAIDの使い方の一例。
二つのディスクを一つのディスクに見せる
…現在のHDDの最大の悩みはバックアップをとれるだけのメディアが同じHDDしか存在してないほど大容量だってコト
そこで使われるのがRAIDと呼ばれる技術なのだ！今どきは安価になってきたから良いかも！
おまえが言うな〜
ディスクの大容量化にも限界ってもんがある！次世代のHDDとしてはいくつか新しい技術によって数倍の容量に！
まだ5年はHDDが主流じゃ
・AFC（反強磁性結合）メディア
ピクシー・ダストと呼ばれる磁性層を使い，400GBまで可能。すでに一部のHDDで使い始めた
・第2世代GMRヘッド
データを記録するヘッド部を改良300GBまで可能。
ふーむ そうか，Dr.Dの脳ミソも大容量にすれば…
いやいや するも何も とっくの昔にHDD化を…
ガパァ
え？
ポチョっとな
！
うわああああ つぐ美ちゃーんっ
ド，ドクターがっ Dr.Dがああっ
ナニ？
何さわいでいるのよーー…って，これ？
ぎゃああっ
ガパァ
うわあああー「電脳化」ってソコまですすんでたのねー
…冗談が通じておらんぞ？
ボクもHDD化するんだい〜
あっあっ
ん―― 案外スナオだったのね ――
TO BE CONTINUED...
…って おぼえてるわけないって？ うーん



これから始めるブロードバンド活用術

ADSL HowTo!! 体験記

実践!! 8MbpsADSL 回線の接続方法

マスコミ報道などで, ADSLを含めたブロードバンド回線の普及に火がつき, 申込み件数が増えすぎて敷設（ふせつ）が間に合わないなどのトラブルが問題になっているようです。これ自体は良くない話題ではあるものの，我が国にも本格的な IT 環境が整いはじめた証（あかし）なのでしょう。さて，今回は ADSL 回線の実際の導入方法を中心に進めていきます。

電話局から 3 Km 以内が ADSL サービスエリアの目安

いつものとおり，最初はADSLに関する技術的な話題から始めましょう。

先月号でお話ししたように，ADSLは，メタル・ケーブル（一般アナログ回線用の電線）に最高 1 MHz 前後までの高周波信号を通して，最高 8 Mbps ものダウンロード通信速度を得ています。

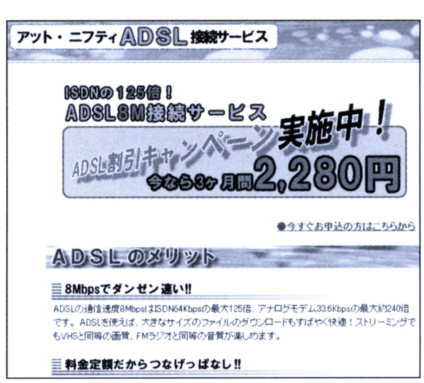
この高周波信号は，皆さんには深夜放送などでお馴染みのAM（中波放送）ラジオで受信している電波とほぼ同じ周波数を持つ信号です。従来のアナログ・モデム方式のパソコン通信は，せいぜい数10kHz 程度の低い周波数信号で通信を行っています。

高周波を使うか低周波を使うかで，送受信のデータ転送速度に比較にならないほどの差がでます。一昔前の電話回線は品質が悪く低周波信号しか扱えないため，それを基準にした通信方式が，アナログ·モデムによるパソコン通信なのです。いまでは，飛躍的に電話回線の品質が向上し高周波信号まで扱えるようになり，ADSLによる高速通信が行える環境になったわけです。

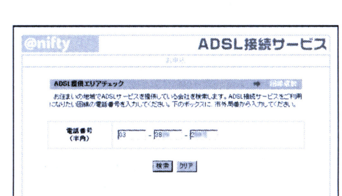
ただし，メタル・ケーブルでADSLの高周波信号をトラブルなく送受信するためには，いくつかの条件があります。

実録 アナログモデム通信から 8MbpsADSL への乗り換えにチャレンジ!! eAccess + Nifty ブロードバンド導入てん末レポート Part ①

昨年夏に「Nifty」（ニフティ）の従量制インターネット接続サービスに加入したところ，ひと月でNTTの通話料も含め，4万円もの料金請求にビックリ!! そこで24時間使い放題で定額制のADSL接続サービスを導入したパソコン・ユーザーの悪戦苦闘の日々を紹介します。ハードウエアの知識にはまったくうといPCビギナーのこの体験は，これからADSLを導入する人には，貴重な資料になるでしょう。なお，導入するADSL回線サービスは「eAccess」（イー・アクセス）の8 Mbpsタイプで，プロバイダーはすでに加入済みの「@Nifty」。今年1月16日に同社のホームページからADSLを申込みました。（取材協力・浅草がらん）

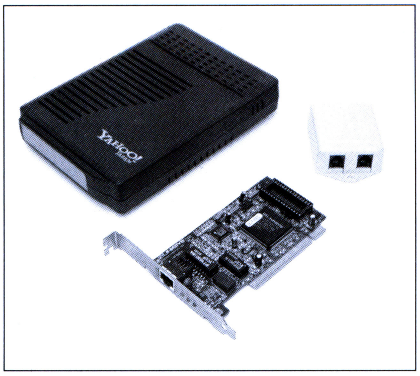


◀アナログ・モデム接続で@Nifty（http://www.nifty.com/）のADSLサービスの申込みページへアクセス。1月時点では，3か月間に限り8MbpsADSLの接続料金が，1,000円割引の月額2,280円となっていた。これまでの高額で低速な接続料金を思えば，まさに天国だ!!

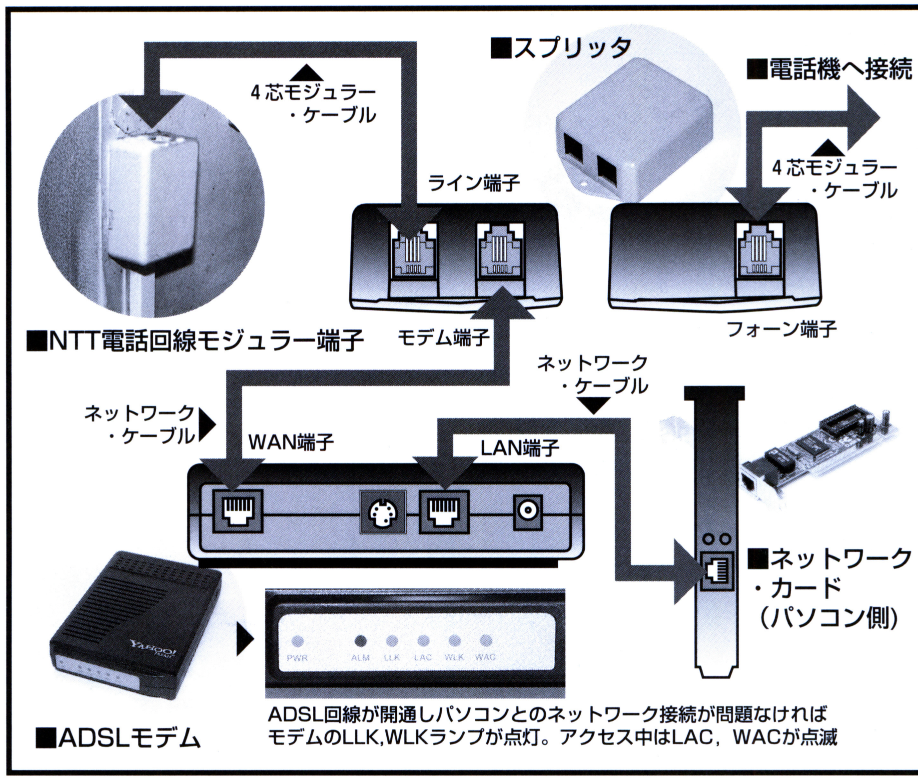


◀申込みの最初のステップは，導入場所がサービスエリア内であるかの確認。電話番号を入力すれば瞬時にわかる

1月16日 インターネット申込み



▲モデムとスプリッタそれにネットワーク・カードは，ADSL導入に欠かせない「三種の神器」だ



▲ＰＣ直結方式のADSL回線導入法（Yahoo! BBの例）

まず，電話局とADSLモデムまでの回線距離ができるだけ短いこと。高周波信号は，低周波よりも外部雑音の影響を受けやすく，その影響を軽減するには，回線距離が短いほど有利になります。

通常，最寄りの電話局のADSL装置と自宅のADSLモデムまでは，3Km程度が限界といわれています。これ以上になると，転送速度が極端に遅くなったり，通信不能におちいることもあります。

この点が，ADSL回線を安全に導入するためのポイントといってもさしつかえないでしょう。

ところで回線距離といっても，正確な距離を自分で計るわけにはいきません。導入場所周辺のマップをもとに，最寄りの電話局（NTT窓口と間違えないように）からのおおよその距離が3Km以内であれば，たいていの場合は大丈夫です。そのあたりは，ADSL業者やNTTの問い合わせ窓口で確認したほうが安心です。

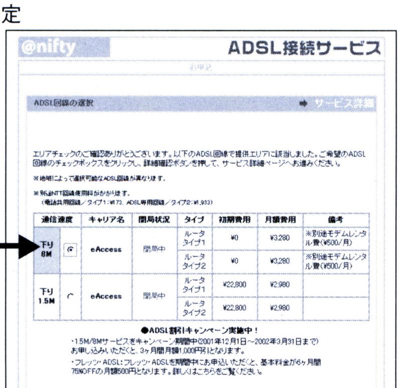
もうひとつの注意点として，事務所や集合住宅などで，特殊な電話交換機が設置してある場合は，ADSLが導入できない場合があります。

この点も事前に調べて，心配ならサポート・センターで相談しましょう。

導入のワンステップはPC直結方式から始めよう

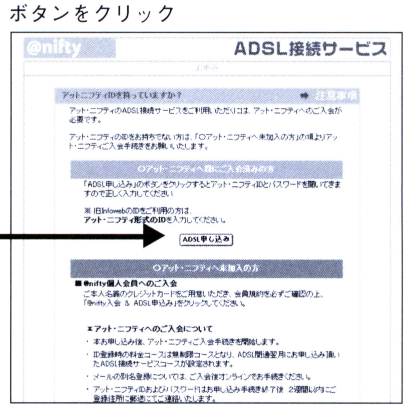
ADSL回線の導入方法は，いくつかの接続方法があることは，先月号で紹介し

1.5Mよりも月額が多少高いが，迷わず「8M」に決定

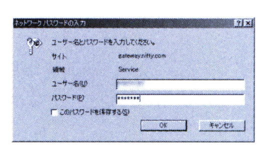


▲@Nifty＋eAccessの組み合わせは，8Mbpsと1.5Mbps（ダウンロード速度）の2タイプから選べる

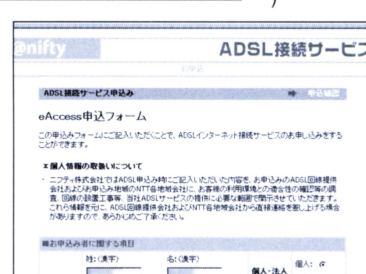
昨年@Niftyに加入済みなので，「ADSL申込み」ボタンをクリック



▲@Niftyへの加入が前提なので，未加入の場合は同時に@Nifty入会手続きを行う



◀@NiftyのIDとパスワードを入力(アナログ接続時と同一)

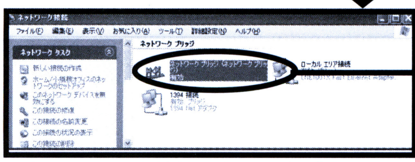


▲ADSL接続サービスの申込みに必要な項目を入力。項目数が多いが慌てずに記入

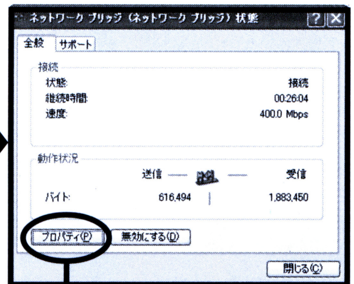
●セキュリティ設定（Windows XP）



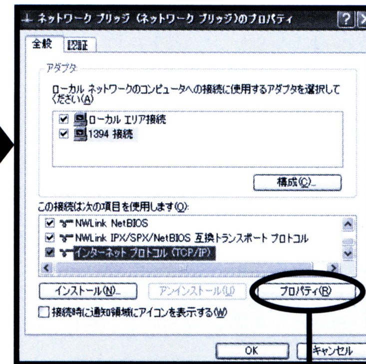
▲まずコントロール・パネルを開き，「ネットワーク接続」を開く



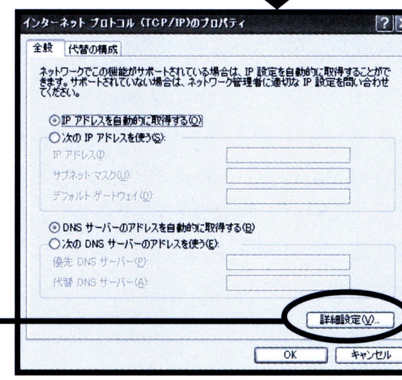
▲ネットワーク接続パネルから「ネットワーク」のアイコンをクリック



▲「全般」タブで，プロパティ・ボタンをクリック



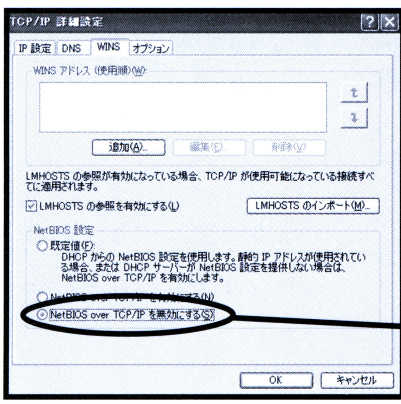
◀「インターネットプロトコル（TCP／IP）」を選びプロパティのボタンをクリック



▲「IPアドレスを自動的に取得する」を選択していることを確認し，詳細設定ボタンをクリック



▲「TCP／IP詳細設定」から「WINS」タブを選ぶ



WINS設定のNetBIOS項目の初期設定「既定値」を「NetBIOS over TCP/IPを無効にする」に変更することで，インターネット上のほかのPCからの不正アクセスを防ぐことが可能

たとおりです。今回は，このなかで最も簡単で低予算で済む接続方法を取り上げてみましょう。

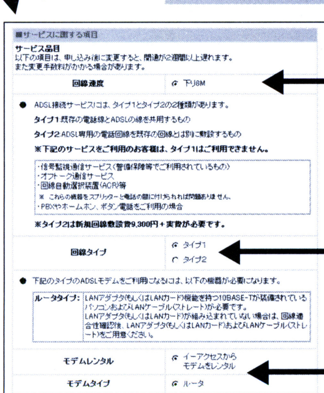
用意する機材は

①ADSLモデム
②ADSLスプリッタ
③ネットワーク・カード
④4ピン・モジュラー・ケーブル×2本
⑤イーサネット・ケーブル×2本

です。このうち①と②は，ほとんどのADSL接続サービス会社では，レンタルが可能です。買い取りシステムもありますが，今後ADSLから光ファイバー回線（FTTH／ファイバー・ツー・ザ・ホーム）への乗り換えを考えると，月額の使用料が数100円ほど高くつくもののレン

実録 「eAccess＋Nifty」ブロードバンド導入てん末レポート①

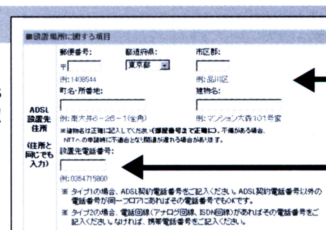
記入項目の重要ポイント



最初に8 Mbpsタイプかを確認しておこう

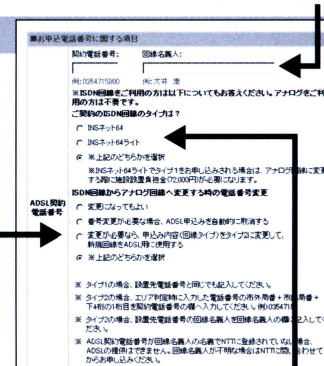
電話とADSL共用のタイプ1，ADSL専用のタイプ2を選択。タイプ1を選んだ

ADSLモデムをレンタルにするかの選択。eAccessではルータ・タイプのモデムになる



ADSLの設置場所の正式な住所を記入

ADSLを導入する回線の電話番号を入力



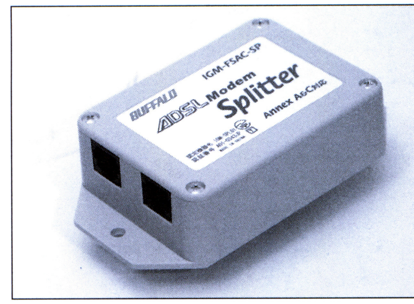
ADSLを導入する電話番号と回線の正式な名義人を記入

ISDN回線からの移行は，電話番号が変更される場合もある。その場合の対応選択項目

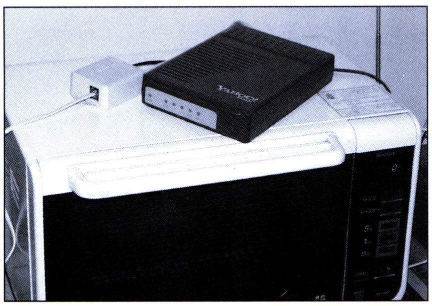
ISDN回線からの乗り換え時に選択。アナログからなのでそのままでよい

▶FAQ 「思うように転送速度が得られない?」場合の対策方法はまずこれから…

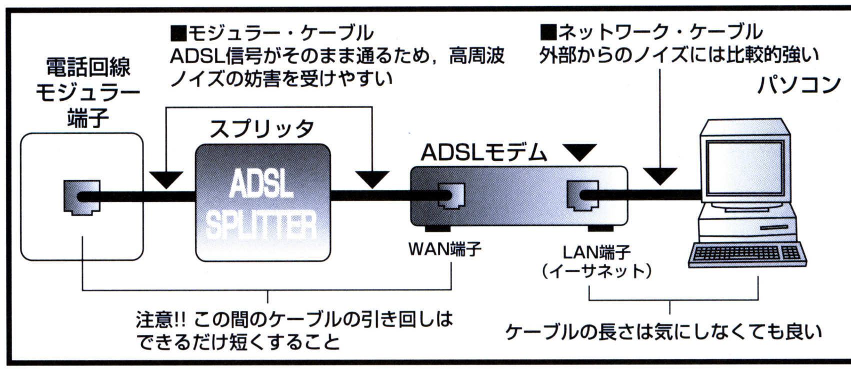
ADSL回線を導入したものの,「思いどおりの通信速度が出ない」,「急に接続不能になる」などのトラブルに見舞われたら，まずは外部ノイズによる影響を疑ってみよう。ADSLで使われている高周波帯のノイズ発生源がモデムやスプリッタ付近にないか調べる。強烈な高周波ノイズを発生する電子レンジなどは，とくに要注意だ。その場合は,ケーブルをノイズ発生源から遠ざけたり,モデムやスプリッタを電話回線コンセントの近くへ設置し,室内のモジュラーケーブル配線を可能な限り短くするなどの対策を試してみよう。



▲電話機に不快なノイズが混入する場合は，メルコ「IGM-FSAC-SP」(1,800円)などのノイズ・フィルタ付きスプリッタが有効



▲電子レンジやテレビなどは，高周波ノイズの発生源にもなる。モデムやスプリッタは，できるだけ離れた場所に設置しよう



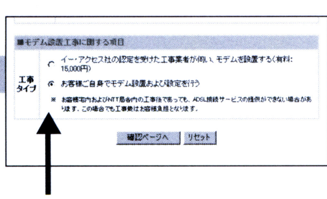
▲ノイズ対策のポイント

タルしたほうが良いと思います。

「eAccess」のようにADSLモデムに，ルータ･タイプをリースしている場合もありますが,内部設定でルータ機能をオフにして通常のADSLモデムとして使えます。モデムとルータが選べる場合は,迷わずルータのほうを選択しておけば,後でパソコンの台数が増えた場合にも,わずかな費用の追加だけで対応できます。

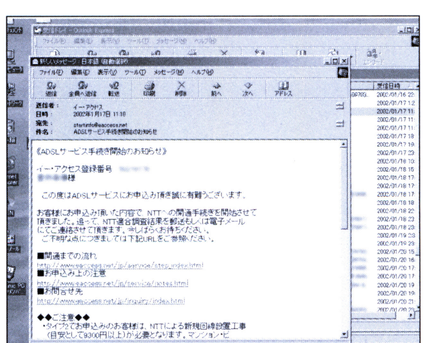
②は，ADSLサービス会社から送られてくる指定のものを用意します。接続するパソコンにネットワーク端子があれば，そのままで問題ないのですが，なければ③のネットワーク･カードをこちらで用意しなければなりません。デスクトップ機はPCIカード，ノートパソコンはLAN PCカードを選びます。どちらも10Base-T／100Base-XTタイプの製品を購入してください。

ケーブル類は，ADSLモデムやスプリッタに付属している場合は，それを使えばよいでしょう。用意されていない場合は,PCショップで必要な本数分

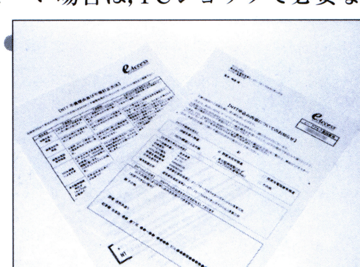


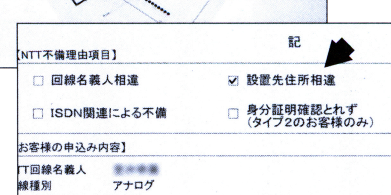
モデムなどのセットアップを業者に頼むことができる。その費用の15,000円を節約するために,自分で工事を行うことにした

※ADSLの申込みは，契約者，連絡先，モデムのレンタル，工事の手配など多くの項目に応えなくてはならない。あせらず一つずつ間違えることなく確実に入力する。とくに電話回線の契約者と住所はNTT電話契約時と同一の内容を記入すること!!



▲ADSL申込みの翌日に，イー・アクセスから内容確認の電子メールが到着。最後までしっかり目を通し，内容を確認したハズなのだが…





▲突然，イー・アクセスから設置住所の記入不備でNTTへの回線確認ができない（矢印）との文書が届いたため，変更手続きを行った

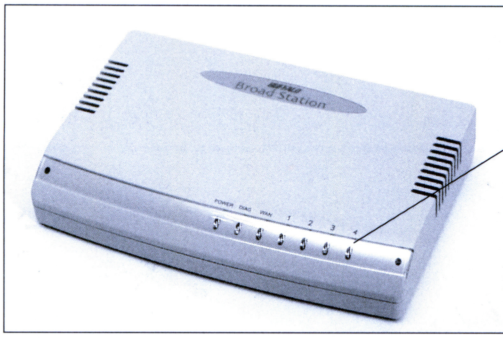
1月17日 受付確認メール着信

1月21日 NTT申請不備連絡

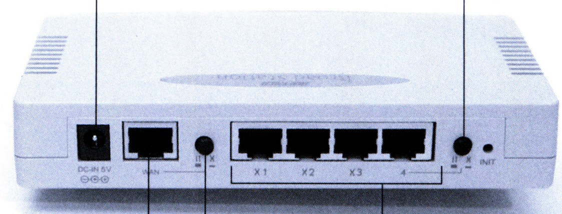
10M/100Mスイッチング4ポートハブ搭載!! 注目のコンパクト&低価格ブロードバンド・ルータ

複数のパソコンからアクセスできたり，外部からの不正アクセスを防ぐファイアウォール機能など，何かと便利なブロードバンド・ルータだが，価格が気になる人にオススメしたいのが，メルコのコンパクト・モデル「BLR-TX4L」(8,500円)。

4ポートの100Base-TX対応LAN端子やスループット7.6Mbpsの高速性能など，上位機種にも決して引けをとらないコストパフォーマンスの良さが特徴だ。



▲手のひらに乗るほどのコンパクトな有線ルータだが，機能や性能は，欠点が見あたらないお得なモデル



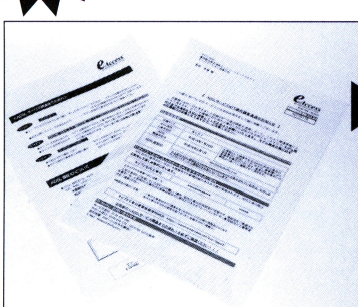
だけ購入します。④は，一般の電話機をつなぐものと同じモジュラー・ケーブルがそのまま流用できます。⑤はイーサネット・ケーブルとも呼ばれ，端子数6ピンのモジュラー端子が両端に着いています。端子とケーブルの太さがひとまわり大きい方が，ネットワーク・ケーブルと覚えておけばよいでしょう。

89ページの接続例は，「Yahoo! BB」のものです。ADSLモデムなどの種類は異なるものの，ほかの会社でも基本的な接続方法は，まったく同じです。最もシンプルなADSL回線の接続方式なので，初心者でも問題ないはずです。

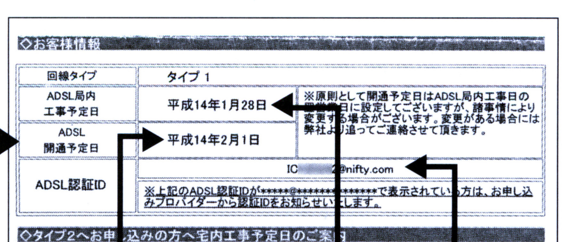
注意ポイントとしては，ADSL接続サービス会社から送られてきたモデムやスプリッタを接続しても，NTTでの回線切替え工事などが完了していなければ，ADSL接続は行えません。利用できるのは，工事前に送られてくる電子メールや文書などに記入されている指定の開通日以降になります。

ちなみに，「Yahoo! BB」でのモデムとパソコン直結方式では，ケーブル類の接続だけでADSL回線が利用可能です。自分のパソコンへの外部からの不正侵入を防ぐためには，90ページにあるWindowsのOS設定を行うだけで極めて簡単です。ただし，ほかのADSL接続サービスでは，モデムやパソコンへの設定が求められる場合もあります。こちらについては，次号で詳しく紹介します。　**(文・福場龍夫)**

実録 「eAccess＋Nifty」ブロードバンド導入てん末レポート①



▲正式にNTT適合審査が行えた内容の文書が，イー・アクセスから郵送されてきた



◇お客様情報

回線タイプ	タイプ 1	
ADSL局内工事予定日	平成14年1月28日	※原則として開通予定日はADSL局内工事日の翌営業日に設定してございますが、諸事情により変更する場合がございます。変更がある場合には弊社より追ってご連絡させて頂きます。
ADSL開通予定日	平成14年2月1日	
ADSL認証ID	IC[redacted]2@nifty.com	
	※上記のADSL認証IDが*****@*************で表示されている方は、お申し込みプロバイダーから認証IDをお知らせいたします。	

◇タイプ2へお申し込みの方へ宅内工事予定日のご案内

ADSL局内工事予定日と同日にお客様宅での新規回線引き込み工事が実施されます。宅内工事の際は

ADSLのNTT局内工事は，1月28日に決定!!

この書面に記入されているADSL認証IDは，ADSLにアクセスする場合に必ず必要。@NiftyのIDとは別もの

イー・アクセスの工事も1月いっぱいに済み，ADSLの開通は2月1日から



▲開通日の前日の夕刻にイー・アクセスから届いたADSLモデム(ルータ・タイプ)とスプリッタ **(以降，次号に続く)**

1月25日 NTT適合通過連絡 → **1月31日** ADSLモデム到着

もっと身近にプログラミング

チャレンジ!!

オリジナル・プログラム言語の制作！

その3 ブロック構造の条件判定文を実装する

文 まかべひろし

CD屋さんをふらっと覗いたら，あまりに懐かしいパッケージが目に止まりました。「北海道連鎖殺人・オホーツクに消ゆ」のサントラです。思わず衝動買いし，原稿執筆のBGMとさせていただきました。iModeでゲームが遊べると聞き，携帯変えようか，なんて本気で悩んでます。

絵 桜部のとす

来ました・・・今年もアレが。花粉です，花粉。ホントツライのですよ～。
今コレ書いてる時ももう目が痒いです。布団とかも外に干せなくなっちゃうし，
はやく花粉シーズン終われ～

catlikeホームページ ▶ http://catlike.marchen.to/

プログラム言語に絶対不可欠なのは，計算処理と，判断処理。今回は「IF文」でおなじみの判断処理を言語システムに実装します。さぁ，オリジナル制作は早くも折り返し地点に突入だ！

さくらの季節到来です～

新学期到来！　この季節になると，不思議と心がウキウキしてしまいますよね。待っているのはきっと，新しい出会い。新しい環境。節目の時期というのはやっぱり，嬉しいことの方がずっとずっと多いと思うのですよ。幸せはやっぱり未来にあるものですもん。何事もポジティブ・シンキング！

さくらの鮮やかな色に心を奪われる毎日は，そんな幸せな気持ちでいっぱいなのです。

条件判定命令の構文を決めよう！

例によって，本編にまったく関係ない，いつもの前振りに気分を良くしたところで，早くも，「オリジナル言語の制作」その3なのです。前回，ひとつ目の山場である演算ルーチンを実装しましたので，今回は軽いものをやっていきたいところですね。さてさて，何が残っているのかな。仕様草案を見て確認してみましょうか。

ベーマガ・マクロプロセッサ（仮）略称「BMP」仕様草案

- ★整数値を格納できる変数システム
- ★演算処理は四則演算対応
- ・条件分岐にIF文。もちろん入れ子対応
- ・BASICライクにGOTO命令でジャンプ
- ・サブルーチン処理用にGOSUB/RETURN
- ・文字列変数や配列も欲しいところ
- ★デバッグ用に，文字列を画面出力
- ・マクロ内にコメントを書けるようにする
- ★は，実装済み

あらら。どれも大変そうですねえ。しいて言うならGOTO命令によるジャンプの実装でしょうか。でも，プログラムを組むということを考えると，ジャンプを実装しても，条件分岐を作らなければ一本道には変わりません。同じ意味で，GOSUB/RETURNも現状での実装はまだ無意味と言えるでしょう。ここは，大変そうに思えますが，先に条件分岐の命令を実装することにしていきます。いわゆる「IF文」ってやつですね。こいつさえできれば，上から下へのバッチ的なプログラムも，条件次第で変化がつけられます。GOTO/GOSUB命令との相性もばっちしです。そして何より，

面倒なところから作っていくと，次回以降が楽な気がするじゃないですか。いや，気がする，だけなのはわかっているんですけれど。

ともあれ，今回はIF文の実装です。BASICやC言語の形だと，IF文は次のような構文ですよね。

IF 条件式A THEN **条件式Aが真の時の処理** ELSE IF **条件式B** THEN **条件式Bが真の時の処理** ELSE **どの条件式も真でなかった時の処理**

THENとかELSEとかの予約語はともかく，これを見ると必要そうなのは，条件式と処理ブロックへの移行部分，そして新たな条件式を探し出すためのキーワードということになりそうです。BASICの場合は，行番号でブロックを管理することができましたが，BMPに行番号というものは存在しません。「ここからここまでが，真の時に処理する内容だよ」というのを明確にする必要があります。また，C言語やPerl等の言語では，条件式の終了キーワードであるTHENを使わないでも問題が出ていないことを考えると，このキーワードは必要不可欠ではないと言えます。

そこで，BMPのIF文は，以下のような構文にしたいと思います。

:IF (**条件式A**) { **条件式Aが真の時の処理** }ELSIF (**条件式B**) { **条件式Bが真の時の処理** }ELSE {**どの条件式も真でなかった時の処理** }

IFの前に「:」が付いているのは，BMP書式のお約束。「ELSE IF」が「ELSIF」に変わったのは，プログラムの解析処理を少しでも簡単にするための工夫です。他の言語の書式から比べても，それほど逸脱しているわけではないので，この仕様で問題はないはずでしょう。

では，実際にこのIF文を動かすためのマクロファイルを作ってみましょう。ちなみに，命令をチェックする時のサンプルですが，最初はできるだけ簡単に，チェック時はできるだけ意地悪なサンプルを作るのが，ミスを残さないポイントと言えます。ま，まずは素直に書いてみることにしましょう。

≪リスト1≫ macro.txt

```
:$a = 1
:IF($a == 1) {
        :PRINT $a
}
```

非常に単純。これで画面に「1」と表示されればOKなわけです。とは言っても，IF文の部分をそのまま飛ばしたら，前回のプログラムでも「1」と表示されてしまうので，ちゃんとIF文を実装した上での話になるのは当たり前。それがうまくいったら，$aに0を代入して，何も表示されないように作っていかなきゃならないのです。

IF文単体の処理・・あれっ，意外に？

そんなわけでIF文のプログラム。まずは「:」の後に，「IF」と続いているかどうかの判定です。これは前回のプログラムの「PRINT」判定部分をそのまま持ってくればOKでしょう。

≪リスト2≫ 命令の増やし方

```
if ( strcmpi( szCmd, "PRINT" ) == 0 ) {
        // PRINTの処理
} else if ( strcmpi( szCmd, "IF" ) == 0 ) {
        // IFの処理
}
```

これから命令を増やしていくときも，同じ要領でやっていけばいいのです。簡単簡単っ，と。

「IF」の次に続くのは，やってきました，条件式。サンプルの「$a == 1」だけなら，比較演算子「==」の左右の値を得て比べればいいだけのように思えますが，例えば条件式には「$a > 2 * 3 + 1」のように計算式が入ることも考えられます。それどころか，「10 <= $a && $a <= 20」10〜20の範囲内に$aがあるか，という複数の条件式が来るかもしれません。

さて，これだけ恐怖を煽っておいてなんですが，この判定式，このように考えることもできます。

「==」や「>」などの比較演算子，「&&」(ANDの意) や「¦¦」(ORの意) などの二項演算子も，四則演算子と同じように「計算の一部」と考えられます。これらの処理の優先順位は，四則演算子の「+」や「-」よりもさらに低く設定します。順番的には，

- **かっこ**
- **通常の四則演算子（+ - * /）**
- **比較演算子（== > < != など）**
- **二項演算子（&& ¦¦）**



こんな感じですね。で，演算処理の中に，これらに対応するコードを入れていくわけなのですが，実際の計算はどういう風にやるべきでしょう。例えば，「1 == 2」の答えはいくつだ，なんて言われても，数値で答えられないから困っちゃいますよね。言葉なら，答えは「正しい」か「間違っている」かですけれど。

さて，ここで，ひとつ知っておいて欲しいことがあります。言語と言えどプログラムです。すべてはコンピュータ上で動く，数値判定のカタマリでしかありません。言語システムにおける条件判定も，もちろん数値であらわされていて，それは，「条件判定式の結果が0か，そうでないか」で判定されています。しかも，普通に考えると「正しいか，正しくないか」という意味でとってしまいがちですが，多くの言語では「0であれば間違っている，それ以外は間違っていない（＝正しい）」と処理しています。このようになった経歴はわかりませんが，とりあえず，この方式にBMPも倣ってみることにしましょう。

そうなると，「1 == 2」の答えは「0」と言えますね。「1 < 2」であれば答えは「0」以外ならなんでも良いのですが，ここでは「0」を全ビット反転した形・・「-1」を答えにしておきます。

二項演算子についても，同じように考えます。「1 < 2 && 3 == 3」の場合は，「&&」の左右をまず計算し「-1 && -1」とし，どちらも「0以外」であれば「-1」を返すようにすればい

いのです。「1＜2¦¦3＝＝4」のように「¦¦」の場合も同様で，「-1 ¦¦ 0」という計算結果が出たら，どちらかが「0以外」であれば「-1」を返します。

こうしてすべての演算子を含めた条件式を計算した結果，「0以外」が返ってくれば，その条件は成立することになるわけです。

実は，前回掲載した演算ルーチンには，この処理がすでに含まれています。つまり，条件式を丸々渡してやると，その答えが「0」か「0以外」かで帰ってくるようになっているのです。

そうとわかれば話は簡単。条件式を処理して，「正しい（0以外）」なら「{」まで処理を進め，正しくないなら「}」まで進めればいいのです。

「じゃあこれは？ それ以外は？」

結構さくっと，IF文単体の処理ができました。さあ，残りの仕様はELSIF，ELSE部分です。まずは，正しい処理ブロックに入るところから考えていきます。これは，IFが成り立たなかった場合に，ELSIFやELSEが続いているかどうかを確認して処理をすればいいのです。ただし，ELSIFが何個続くかわからないので，ELSIFが来た場合のみ条件式のチェックに戻るようにする必要があります。

リスト3は，IF命令が来た後，正しいブロックに入るまでを記述したものです。入る処理があれば出る処理もあります。マクロの解析処理において，この部分だけは他とは違う作り方をしなくてはいけません。なぜなら，他は「:」が頭についた「命令」であるのに対し，ELSIF，ELSEは，なんの前触れもなくあらわれる言葉だからです。

「え，IFの続きとして処理すればいいんじゃないの？」

なんて思いがちではありますが，

IF（**条件式**）{ **処理A** } ELSE { **処理B** }

という形を思い浮かべてください。条件式が否であれば処理Aは実行されず，処理Bに移行しますが，条件式が真であった場合，処理Aが実行されたあと「ELSE」が来てしまうのです。正確には，「}」が来た後には，「ELSIF」が来る，「ELSE」が来る，何も来ない，という3パターンが考えられます。

となると，命令の頭である「:」を判定する部分と平行に，「}」が来たかどうかを確認しなきゃいけませんね。

実際のプログラムはこんな感じ。「}」が来た後に「ELS」の文字があるかを確認し，あった場合は，その後のカッコにくくられた部分は無視してしまうようにします。

忘れちゃいけないのが，続く「ELS」の数はひとつとは限らないこと。IF～ELSIF～ELSIF～ELSE，なんて形も考えられますよね。ですから，1ブロック飛ばしても安心せずに，再び「ELS」が来ているかどうかをちゃんと確認しなくてはいけません。ここでは，while(1)がその役目を果たしています。

これで一応，仕様は満たしたことになります。おお，IF文完成！？いやいや，まだ「意地悪チェック」が残ってるんです。

入れ子のIF文に対応しちゃお

次のようなマクロファイルを用意して，今作った言語システムに通してみてください。見てわかるとおり，意地悪ってほどのものではありません，IF文を使えるのならば，当然こんな入り組んだ形もありうる話ですよね。あ，ちなみに，PRINT命令は文字列が表示できるもの

≪リスト3≫ IF命令

```
} else if (strcmpi( szCmd,"IF") == 0){
        //IFの処理
        while(1){
                int nMath;
                nPnt+=GetIntFromString( (char*)&p[nPnt], &nMath);
                if(nMath!=0){
                        // okなので { まで進めよう
                        while( p[nPnt] != '{' ) { nPnt++; } nPnt++;
                        break;
                } else {
                        // ダメなので } まで進めよう
                        int nOldPnt;
                        while( p[nPnt] != '}') { nPnt++;} nPnt++;
                        nOldPnt = nPnt;
                        nPnt += SkipSpaceFromMemory(&p[nPnt]);
                        if (strnicmp((char*)&p[nPnt],"elsif", strlen("elsif"))==0){
                                nPnt+=strlen("elsif");
                        }else if(strnicmp((char*)&p[nPnt],"else",strlen("else"))==0){
                                nPnt += SkipString( &p[nPnt], "{" );
                                nPnt++;
                                break;
                        } else {
                                nPnt = nOldPnt;
                                break;
                        }
                }
        }
}
```

≪リスト4≫ IFブロックを正しく抜ける

```
//解析ループです
while(nPnt<nSize){
        if(IsKanji(p[nPnt])){
                //SHIFT-JIS漢字コードなので2byte進める
                nPnt += 2;
        }else{
                if(p[nPnt]== ':'){
                        //コマンドの処理
                        //（省略）
                }else if (p[nPnt]== '}'){
                        // 「}」が来た後
                        int nOldPnt;
                        nPnt++;
                        while( 1 ) {
                                nOldPnt = nPnt;
                                nPnt += SkipSpaceFromMemory( &p[nPnt] );
                        if ( strnicmp( (char*)&p[nPnt],"els", strlen("els"))
==0){
                                        // 次の「}」が来るまでルー
プ
                                        while( p[nPnt] != '}' ) {
nPnt++;}
                                        nPnt++;
                                } else {
                                        nPnt = nOldPnt;
                                        break;
                                }
                        }
                } else {
                        // コマンドでないので1byte進める
                        nPnt++;
                }
        }
  }
```

としますね。

普通に見ると，「(a)ELSIF文成立」と出て終了するべきですが，「(b)ELSIF文成立」と表示されてしまいます。これは，IF不成立後の飛ばし処理に問題があり，ひとつ目の条件式「$a == 7」が成立しなかったので，次の「}」まで進めようとした際，「$a == 8」まで行かずに途中で引っかかってしまうのが原因です。

これを回避するためには，途中でIFブロックがあることを考慮し，「{」と「}」の数を調べていけばいいことになります。具体的には，カウンタを用意し，「{」の時は+1，「}」の時は-1をするようにして，カウンタが0に戻るまでループさせてください。

これらを反映し，PRINT命令も拡張した，今回の完成プログラムが**リスト6**です。前のプログラムに追加する形になりますので，前回の記事もあわせて参照くださいね。

≪リスト5≫ macro.txt

```
:PRINT "ここから始まります"
:$a = 8
:$b = 6
:IF( $a == 7 ) {
		:PRINT "(a)IF文成立"
		:IF( $b == 4 ) {
				:PRINT "(b)IF文成立"
		} ELSIF ( $b == 5 ) {
				:PRINT "(b)ELSIF文成立"
		} ELSE {
				:PRINT "(b)ELSE文成立"
		}
} ELSIF ( $a == 8 ) {
		:PRINT "(a)ELSIF文成立"
} ELSE {
		:PRINT "(a)ELSE文成立"
}
```

≪リスト6≫ script.c（差分）

```
//追加関数
int SkipString( unsigned char *p, char *sz ) {
	int nCnt = 0, nFlg = 1, nQuoteFlg = 0;
	unsigned int i;
	while(nFlg){
		if (!nQuoteFlg ) {
			for ( i = 0; i < strlen( sz ); i++ ) {
				if ( p[nCnt] == (unsigned char)sz[i] ) { nFlg = 0; }
			}
		}
		if ( nFlg ) {
			if ( IsKanji( p[nCnt] ) ) {
				nCnt += 2;
			} else {				nCnt++;
			}
		}
	}
	return nCnt;
}

// プログラムはここから始まる
void main() {
	FILE *fp; int nSize;
	unsigned char *p;
	int nPnt = 0;

	// 初期化
	ValueSysInit();

	// マクロファイルをメモリに読み込む
	fp = fopen( "macro.txt", "rb" );
	fseek( fp, 0, SEEK_END ); nSize = ftell( fp ); fseek( fp, 0, SEEK_SET );
	p = ( unsigned char* )malloc( nSize + 1);
	fread( p, 1, nSize, fp );
	fclose( fp );

	// 読んだデータの終端に¥0をつける
	p[ nSize ] = '¥0';

	// interpriter //////////////////////////////////////////////////////////////////
	while( nPnt < nSize ) {
		if ( IsKanji( p[nPnt] ) ) {
			// SHIFT-JIS漢字コードなので2byte進める
			nPnt += 2;
		} else {
			if ( p[nPnt] == ':' ) {
// コマンドマーク有
				nPnt++;
				if ( p[nPnt] == '$' ) {
					// 変数代入
					char szValueName[32];
					int nValue;
					nPnt++;
					nPnt += ValueSysGetValueName( (char*)&p[nPnt], szValueName, sizeof(szValueName) );
					while( p[nPnt] != '=' ) { nPnt++; } nPnt++;
					nPnt += GetIntFromString( (char*)&p[nPnt], &nValue );
					ValueSysSetData( szValueName, nValue );
				} else {
					// コマンド処理
					char szCmd[32]; int i = 0;
					while( ( 'A' <= p[nPnt] && p[nPnt] <= 'Z' )
						|| ( 'a' <= p[nPnt] && p[nPnt] <= 'z' ) ) {
						szCmd[i] = p[nPnt]; nPnt++; i++;
```

→ つぎのページに続く

```
                    }
                    szCmd[i] = '¥0';
                    if ( strcmpi( szCmd, "PRINT" ) == 0 ) {
                        int nAns;
                        nPnt += SkipSpaceFromMemory( &p[nPnt] );
                        if ( p[nPnt] == '"' ) {
                            // 文字列表示
                            nPnt++;
                            while( p[nPnt] != '"' ) {
                                printf( "%c", p[nPnt] ); nPnt++;
                            }
                            printf( "¥n" ); nPnt++;
                        } else {
                            nPnt += GetIntFromString( (char*)&p[nPnt], &nAns );
                            printf( "%d¥n", nAns );
                        }
                    } else if ( strcmpi( szCmd, "IF" ) == 0 ) {
                        while( 1 ) {
                            int nMath;
                            nPnt += GetIntFromString( (char*)&p[nPnt], &nMath );
                            if ( nMath != 0 ) {
                                // ok.
                                nPnt += SkipString( &p[nPnt], "{" );
                                nPnt++;
                                break;
                            } else {
                                // no.
                                int nCnt = 0, nOldPnt;
                                do {
                                    nPnt += SkipString( &p[nPnt], "{}" );
                                    if ( p[nPnt] == '{' ) { nCnt++; }
                                    if ( p[nPnt] == '}' ) { nCnt--; }
                                    nPnt++;
                                } while( nCnt );
                                nOldPnt = nPnt;
                                nPnt += SkipSpaceFromMemory( &p[nPnt] );
                                if ( strnicmp( (char*)&p[nPnt], "elsif", strlen( "elsif" ) ) == 0 )
{

                                    nPnt += strlen( "elsif" );
                                } else if ( strnicmp( (char*)&p[nPnt], "else", strlen( "else" ) ) ==
 0 ) {

                                    nPnt += SkipString( &p[nPnt], "{" );
                                    nPnt++;
                                    break;
                                } else {
                                    nPnt = nOldPnt;
                                    break;
                                }
                            }
                        }
                    }
                }
            } else if ( p[nPnt] == '}' ) {
                int nOldPnt;
                nPnt++;
                while( 1 ) {
                    nOldPnt = nPnt;
                    nPnt += SkipSpaceFromMemory( &p[nPnt] );
                    if ( strnicmp( (char*)&p[nPnt], "els", strlen( "els" ) ) == 0 ) {
                        // skip..
                        int nCnt = 0;
                        do {
                            nPnt += SkipString( &p[nPnt], "{}" );
                            if ( p[nPnt] == '{' ) { nCnt++; }
                            if ( p[nPnt] == '}' ) { nCnt--; }
                            nPnt++;
                        } while( nCnt );
                    } else {
                        nPnt = nOldPnt;
                        break;
                    }
                }
            } else {
                // コマンドでない ので1byte進める
                nPnt++;
            }
        }
    }
    // 解放
    ValueSysRelease();
    free( p );
}
```

待て、次号！

ついに，IF文も完成しました。上から下に流れるバッチ的なプログラムなら，これでもなんとか組めてしまうのです。となると，次に欲しいのはやっぱり上下移動が自由にできるGOTO文。次回はBMPに，一見簡単そうに見えるジャンプ命令を搭載していきたいと思います。

さあ，今日は，二歩野望に近づいた！！

(無　料)

フリーソフト

HSP完全マスター

Hot Soup Processor

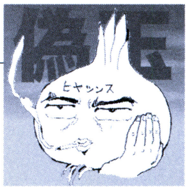
楽々3Dプログラミング！

第3回

監修：おにたま

「カメラとオブジェクト」

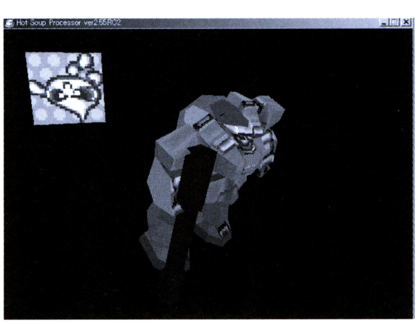
講師：悠黒喧史



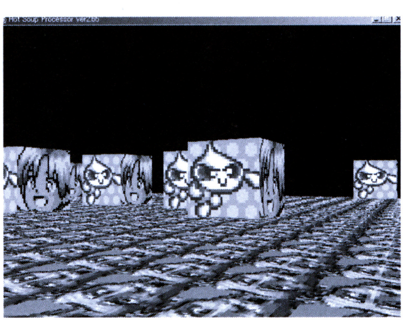
はじめてHSPと出合ったときは面白さのあまりフリーソフトなのにカンパ(千円)。
その後，好きが高じてHSPを書籍出版までこぎつけた。HSP Usherを自称する。

今回のポイント

注視点モード



自機からの視点



カメラポジション

HGIMGにはカメラが存在します。テレビ放送や監視に使われるモニタカメラのようなものを想像してください。実は，これまで意識せずに済ませてきましたが，今までの3Dモデルを表示している画面もすべて，このカメラからの視点になっていたのです。

カメラは，特に何も指定がない場合には，HGIMGワールドの座標(0,0,4)にあり，まっすぐ-Zの方向を向いているのですが，専用の命令がいくつか用意されており，座標と角度を自由に変更することができます。

カメラを動かして視点を変えてみよう

カメラ座標の指定には，selcpos命令を使います。

指定方法は，オブジェクトと同様で，次のように指定します。

スクリプト記述例

```
selcpos :objset3 0,-20,60
```

カメラ座標を(0,-20,60)にセットしています。中心座標から，60歩後退して，20歩上昇した位置と考えるとわかりやすいでしょう。

ちなみにobjset3命令は整数値での指定ですが，より細かな数値(小数値)で指定したいときには，objsetf3命令を使うことができます。

カメラ角度の指定には，selcang命令を使います。

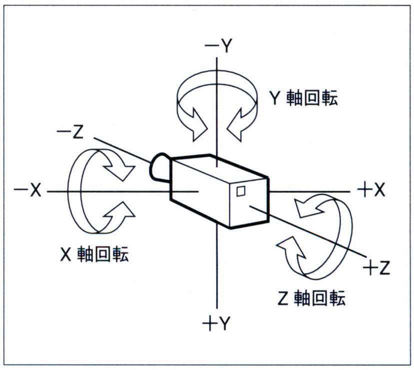
スクリプト記述例

```
selcang :objsetf3 0.5f,0.0f,0.0f
```

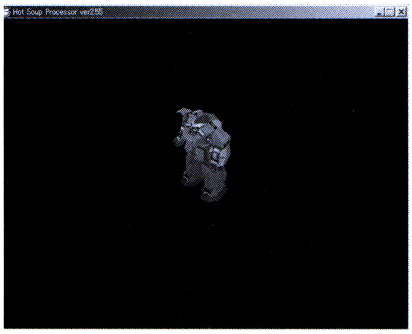
カメラ角度をX軸に約30度(0.5)傾けます。カメラをやや下に向け，見下ろす感じです。

角度の数値は，オブジェクトでの指定と同様にラジアンで指定します。

カメラのX,Y,Z軸回転図

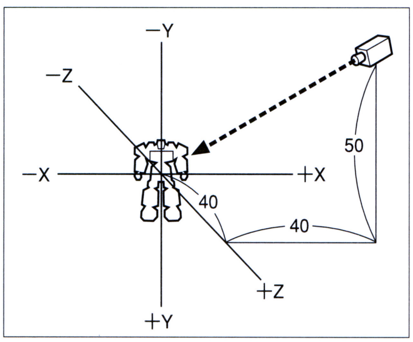


リスト1の実行画面



リスト1の解説

カメラは，右斜め上から，オブジェクトを見下ろす姿勢になっています。



目の前に何もないと，視点が変更されたかどうかもわからないので，MXモデルのオブジェクトを作成して，中心点に置いています。

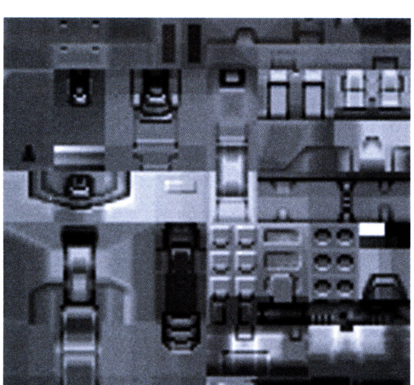
モデルデータは，HSP2.55に同梱されているサンプルデータ(**sample¥hgimg**フォルダにある**rb_010.mx**と**rb_001.bmp**)を使用しました。

MXモデルデータ

rb_010.mx

MXモデルに貼るテクスチャ

rb_001.mx
128×128ドット



カメラ視点の設定は，**selcpos**と**selcang**命令のある2行だけです。

objset命令で指定している数値を変えて，カメラ視点がどう変化するか，いろいろ試してみると良いでしょう。

使用した命令

elcpos　カメラ座標をアクセス対象にする
selcang　カメラ角度をアクセス対象にする

《リスト1》カメラ視点の変更

```
#include "hgimg.as"

        hgini                                   ; HGIMGの初期化
; ---------------------------------------------------------------------------
        mxload "rb_010"                         ; MXﾓﾃﾞﾙﾌｧｲﾙ読込み
; ---------------------------------------------------------------------------
        regobj ob,stat                          ; ｵﾌﾞｼﾞｪｸﾄ作成
; ---------------------------------------------------------------------------
        selcpos :objset3 40,-50,40              ; ｶﾒﾗ座標(40,-50,40)
        selcang :objsetf3 0.78f,0.78f,0.0f      ; ｶﾒﾗ角度(45度,45度,0度)
; ---------------------------------------------------------------------------
        gsel 0
        hgdraw                                  ; 全ｵﾌﾞｼﾞｪｸﾄ描画処理
        hgsync 10                               ; ｳｪｲﾄ
        stop
```

カメラモード

カメラ視点には，**ノーマルモード**と**注視点モード**の二つのモードがあります。通常，何も指定されていない状態では，ノーマルモードになっています。

ノーマルモード

《リスト2》注視点モードの用例

```
#include "hgimg.as"

        hgini                                   ; HGIMGの初期化
; ---------------------------------------------------------------------------
        setsizef 5.0f,5.0f                      ; 立方体ﾓﾃﾞﾙのXYｻｲｽﾞを指定
        setuv 0,0,31,31                         ; ﾃｸｽﾁｬ画像の1面範囲
        addbox md                               ; 立方体ﾓﾃﾞﾙの登録(mdにﾓﾃﾞﾙIDが返る)
        texload "bxtx.bmp"                      ; ﾃｸｽﾁｬｰの読込み
        regobj ob,md                            ; ｵﾌﾞｼﾞｪｸﾄ作成
; ---------------------------------------------------------------------------
        mxload "rb_010"                         ; MXﾓﾃﾞﾙﾌｧｲﾙ読込み
        regobj ob,stat                          ; ｵﾌﾞｼﾞｪｸﾄ作成
        selpos ob :objset3 0,0,-30              ; ちょっと距離を離す
; ---------------------------------------------------------------------------
        cammode CAM_MODE_LOOKAT                 ; ｶﾒﾗを注視点ﾓｰﾄﾞにする
        selcpos :objset3 0,0,30                 ; ｶﾒﾗ位置 30歩後退
; ---------------------------------------------------------------------------
        gsel 0
*main
        selang ob                               ; MOC角度にｱｸｾｽ

        k=-1                                    ; ﾃﾝｷｰ入力ﾁｪｯｸ
        repeat 10                               ;
                getkey a,96+cnt                 ;
                if a=1 :k=cnt :break            ;
        loop                                    ;

        if k=8 :objaddf1 0,-0.1f                ; X軸回転
        if k=2 :objaddf1 0,0.1f                 ;
        if k=4 :objaddf1 1,-0.1f                ; Y軸回転
        if k=6 :objaddf1 1,0.1f                 ;
        if k=1 :objaddf1 2,-0.1f                ; Z軸回転
        if k=0 :objaddf1 2,0.1f                 ;

        selpos ob                               ; MOC座標にｱｸｾｽ

        stick ky,79                             ; ｷｰｽｷｬﾝ

        if ky&1 :objaddf1 0,-0.5f               ; X軸移動
        if ky&4 :objaddf1 0,0.5f                ;
        if ky=2 :objaddf1 1,-0.5f               ; Y軸移動
        if ky=8 :objaddf1 1,0.5f                ;
        if ky=66 :objaddf1 2,-0.5f              ; Z軸移動
        if ky=72 :objaddf1 2,0.5f               ;

        objgetfv p                              ; MOC座標を変数pにｹﾞｯﾄ
        selcint :objsetfv p                     ; 変数pの座標値を注視点にｾｯﾄ

        hgdraw                                  ; 全ｵﾌﾞｼﾞｪｸﾄ描画処理
        hgsync 10                               ; ｳｪｲﾄ

        if ky!128 :goto *main                   ; [Esc]が押されるまでﾙｰﾌﾟ

        hgbye                                   ; HGIMGを解放
```

ノーマルモードでは，座標と角度を指定し，カメラ視点を設定します。

スクリプト記述例

cammode CAM_MODE_NORMAL

注視点モード

注視点モードとは，カメラが常にある一点を見つめるようにするモードです。カメラの座標はノーマルモードと同じように指定しますが，カメラの角度は，注視する座標を指定することで，自動的に決まるようになります。

スクリプト記述例

cammode CAM_MODE_LOOKAT

注視点モードにすると，selcintという命令が有効になります(ノーマルモードでは，この命令は記述しても無視されます)。selcint命令は，注視する座標を指定するための命令で，次のように指定します。

スクリプト記述例

selcint :objset3 10,20,40

カメラの注視点を(10,20,40)にセットします。カメラをどの座標に移動しても，カメラの角度は常にこの座標(10,20,40)を見つめるように自動的に設定されます。

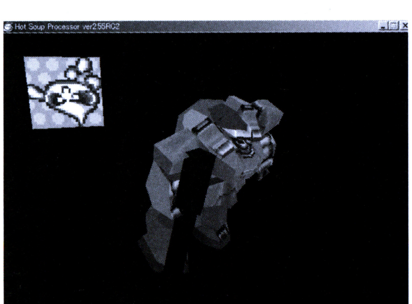
立方体に貼り付けるテクスチャ

bxtx.bmp

32×96ドット



リスト2の実行画面



カーソルの↑↓←→で上下左右，Ctrlを押しながらの↑↓で前後に移動します。

テンキーの4 6でY軸回転，8 2でX軸回転，1 0でZ軸回転をします。

Escでプログラムを終了します。

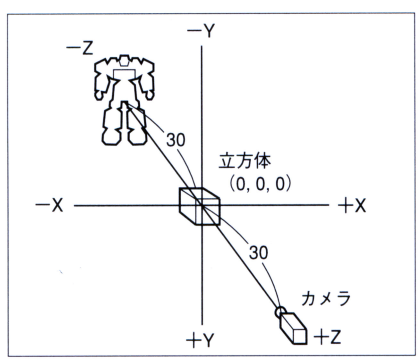
リスト2の解説

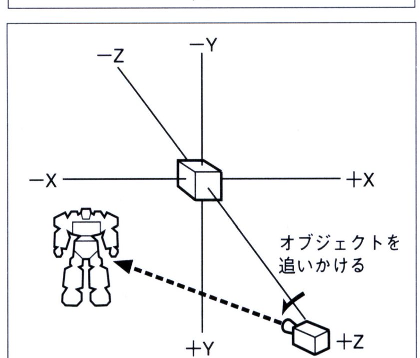
移動するオブジェクトを注視点に設定することで，カメラが，動くオブジェクトを追いかけるというサンプルプログラムです。

MXモデルのオブジェクトをカーソルキーで移動，カメラの注視点を常にこのオブジェクトの座標に設定します。オブジェクトを追いかける動きがわかりやすいように，中心点(0,0,0)に動かない立方体モデルを置いています。

注視点をオブジェクトに指定する方法は，まず，selpos命令でオブジェクトのMOC座標をアクセス対象にし，objgetfv命令で座標値を変数p.0〜2に代入。次にselcint命令で注視点をアクセス対象にし，objsetfv命令で変数p.0〜2の内容(座標値)をセットしています。

オブジェクトを色々な方向に移動させて，視点(画面)の動きがどのように変化するか，試してみてください。





使用した命令

cammode	カメラモードの設定
selcint	注視点座標をアクセス対象にする

自機からのカメラ視点について

レーシングゲームのドライバーの視点など，プレイヤーが操作する自機オブジェクトから見た光景をカメラ視点にしたい場合，注意すべきポイントがあります。

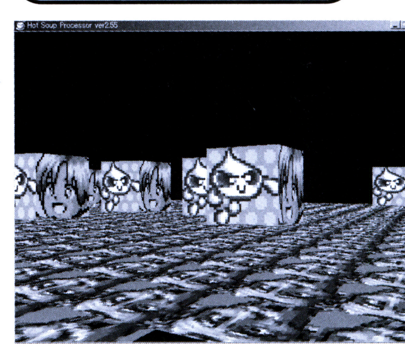
それは，HGIMGではシーン=オブジェクト×カメラという計算で画面(シーン)を表示しているため，オブジェクトの角度をカメラ用に変換するには，逆数にしないといけないということです。

記述例

```
elang ob :objgetfv an ；自機オブジェクト角度
fadd an.1,3.14f
fvmul an, -1.0f, -1.0f, -1.0f
selcang :objsetfv an ；自機視点(角度)
```

カメラ座標は，自機オブジェクトの座標値をそのままコピーすれば良いのですが，カメラ角度には，自機オブジェクトの角度値にマイナスをかけ，逆数にしてセットします。

リスト3の実行画面



カーソルの←→で左右に回転，↑↓で前進・後退します。

Escでプログラムを終了します。

リスト3の解説

addmeshは，複数に分割された面をもつプレート(板)モデルを作成する命令です。このプレートモデルを床として配置します。

次に，立方体モデルを複数作成し，床の上に乱数で適当な座標に配置します。

MXモデルを自機オブジェクトとして作成します。モデルのスケールが大きくて床からハミ出すので，selscale命令で

《リスト3》自機からの視点

```
#include "hgimg.as"

        hgini                               ; HGIMGの初期化
; -----------------------------------------------------------------------
        setsizef 64.0f,64.0f                ; ﾓﾃﾞﾙのXYｻｲｽﾞを指定
        setuv 0,0,31,31                         ; ﾃｸｽﾁｬの1面範囲
        addmesh mdpl,32,32,16                   ; ﾌﾟﾚｰﾄﾓﾃﾞﾙ(MESH)の登録
; -----------------------------------------------------------------------
        regobj obpl,mdpl                    ; ﾌﾟﾚｰﾄﾓﾃﾞﾙで床を作成
        selang obpl :objsetf1 0,-1.57f      ;
        selpos obpl :objset1 1,2            ;
; -----------------------------------------------------------------------
        setsizef 4.0f,4.0f                  ; ﾓﾃﾞﾙのXYｻｲｽﾞを指定
        setuv 0,0,31,31                         ; ﾃｸｽﾁｬの1面範囲
        addbox mdbx                             ; 立方体ﾓﾃﾞﾙの登録
        texload "bxtx.bmp"                  ; ﾃｸｽﾁｬ画像の読込み
; -----------------------------------------------------------------------
        repeat 16 :rnd z,64 :rnd x,64       ; 立方体を適当に配置
                regobj obbx,mdbx
                selpos obbx :objset3 x-32,0,z-32
        loop
; -----------------------------------------------------------------------
        mxload "rb_010"                     ; MXﾓﾃﾞﾙﾌｧｲﾙ読込み
        regobj obmy,stat                    ; ｵﾌﾞｼﾞｪｸﾄ作成
        selscale obmy :objsetf3 0.2f,0.2f,0.2f ; ﾓﾃﾞﾙｽｹｰﾙを1/5に調整
; -----------------------------------------------------------------------
        gsel 0
*main
        selang obmy :objgetfv an            ; an ← 自機のＭＯＣ角度

        stick ky,31
        if ky&1 : fsub an.1,0.01f           ; 左回転
        if ky&4 : fadd an.1,0.01f           ; 右回転

        objsetfv an                         ; 自機のMOC角度 ← an

        selpos obmy :objgetfv po            ; po ← 自機のMOC座標

        if ky&8 { fvdir an,0.0f,0.0f,-0.05f ; 前進
                fvadd po,an.0,an.1,an.2     ;
        }
        if ky&2 { fvdir an,0.0f,0.0f,0.05f  ; 後退
                fvadd po,an.0,an.1,an.2     ;
        }
        selpos obmy :objsetfv po            ; 自機のMOC座標 ← po

        selcpos :objsetfv po                ; 自機視点(座標)
        selang obmy :objgetfv an            ;
        fadd an.1,3.14f                     ;
        fvmul an,-1.0f,-1.0f,-1.0f          ;
        selcang :objsetfv an                ; 自機視点(角度)

        hgdraw :hgsync 10

        if ky!128 :goto *main               ; [Esc]が押されるまでﾙｰﾌﾟ

        hgbye                               ; HGIMGを解放
```

1/5にサイズを調節しました。

自機オブジェクトの表示がジャマな場合は，次のようにオブジェクトの属性を**非表示指定**にしてみてください。

スクリプト記述例

```
regobj obmy,stat,OBJ_HIDE
```

fvdir命令を利用した移動処理を行なっています(3月号/第2回講座参照)。

selcpos〜selcang命令の5行で，自機視点を設定しています。この5行をコメントアウトして実行すると，自機オブジェクトの　動く様子を外側から見ることもできます。

```
; selcpos :objsetfv po; 自機視点(座標)
; selang obmy :objgetfv an       ;
; fadd an.1,3.14f                ;
; fvmul an,-1.0f,-1.0f,-1.0f     ;
; selcang :objsetfv an   ; 自機視点(角度)
```

自機視点の設定部分をコメントアウト※して実行

※コメント記号";"を行頭に入れて，実行されない行にすること

実行画面



使用した命令

addmesh　分割ポリゴンの板(MESH)を作成

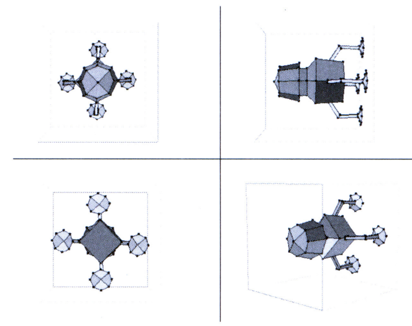
selscale　モデルの拡大・縮小率を設定

fvmul　変数(配列0,1,2)に3つの小数値を一度に乗算

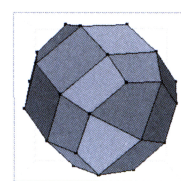
次回からは，実際に3Dゲームを制作しながら，その過程を詳しく解説してゆきます。

3DACEにて制作中のゲーム用モデル

ロケット(自機)



隕石

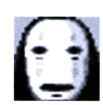


時間のある人は，次回講座用にこれらのモデルを作っておくと良いかもしれません。

大体のサイズが合っていればデザインは異なっててもかまいませんよ。

HGIMG-FAQ

質問



モデルに貼るテクスチャ画像にはBMP(ビットマップ)形式しか使えないのですか？

答え



3Dモデルに貼り付けるテクスチャ画像には，BMP,MAG,JPG形式が使用できます。

BMP形式はRLE圧縮されたものも使えるので，容量を小さくしたい場合には活用しましょう。

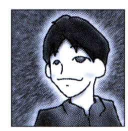
教科「情報」など授業にも使える！

Visual Basic アルゴリズム講座

文・プログラム／東 幸太

第1回 アニメーション効果など「動き」を表現する

この連載は，趣味だけでなく，学校の授業でも使用してもらえるような身近にあるテーマをもとに，さまざまなアルゴリズム（コンピュータ・プログラムを作るときに使われる問題解決方法や考えかた）を解説，Visual BASICで実際にプログラムを動作させていきます。



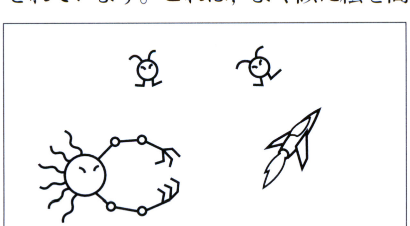
寒さが厳しくなってくると，キーボードを打つ指先が冷えて困る。人間は集中すると脳に血液が回り，指先足先がその分冷えてしまうらしい。発熱体装備のウォーマー付きキーボードが欲しいところ（笑）。

アニメーション効果

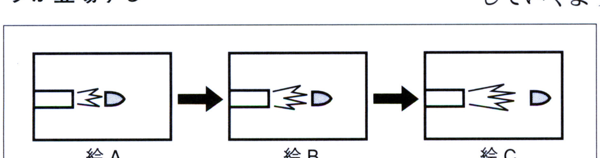
コンピュータゲームでは，ゲームフィールドの中をさまざまなキャラクタが移動していきます（**第1図**）。大半のゲームでは，プレイヤーの視覚と聴覚によってゲームの状況を伝えますから，視覚に対応する画面表示は重要な役割をもっています。画面表示方法が不適切だと，キャラクタが移動しているように見えなかったり，移動しているように見えてもぎこちなく見えたりして，ゲームの面白さに悪影響を与えてしまいます。

これらのキャラクタの移動を表現する手法としてはアニメーション効果が利用されています。これは，よく似た絵を高速に切り替えた場合に，それぞれの絵の違いがなめらかに変化したように見えることを指します。**第2図**のように，よく似た絵を短い時間で切り替えながら表示したとします。実際には，絵Aから絵Bへ，絵Bから絵Cへ，絵が不連続に切り替わっていくのですが，これを人間が見ると，それぞれの絵の違いの不連続さがあまり感じられず，連続してなめらかに変化したように見えます。アニメーションテレビ番組などでは，動きを表現する際によく似た絵を多数用意し，それらを切り替えながら表示することで，実際には不連続な動きをなめらかに見せています。

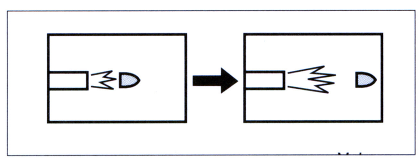
このアニメーション効果を利用することで，不連続な絵しか表示できないコンピュータでも，見た目はなめらかに変化していくように見せることができます。ただし，このアニメーション効果を得るためには，切り替えていく絵の違いが少ないことと，切り替える時間が速いことが要求されます。



《第1図》ゲーム中には多くのキャラクタが登場する



《第2図》よく似た絵を切り替えていくことで，アニメーション効果が得られる



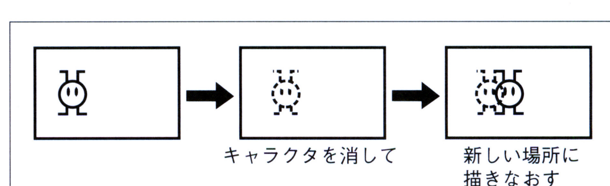
《第3図》絵の違いが大きすぎるとアニメーション効果は得られない

第3図のように，絵を切り替えて表示したとしても，個々の絵の違いが大きすぎると人間には絵が切り替わったように見えてしまい，動きがあるようには見えません。また，絵を切り替える頻度が少なく，たまにしか変化しない場合は，カクカクした動きに見えてしまい，なめらかには見えないのです。アニメーション効果を期待する場合は，これらの点に注意しなければなりません。

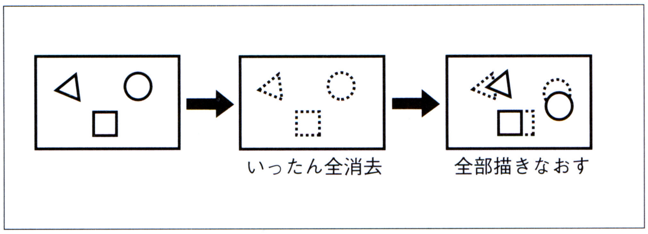
さまざまな表現方法

コンピュータは多彩な画像を表示することができます。当然，さまざまなキャラクタも表示できますが，それらのキャラクタに動きを与えることを考えます。キャラクタにアニメーション効果による動きを与えるためには，いくつかの方法があります。

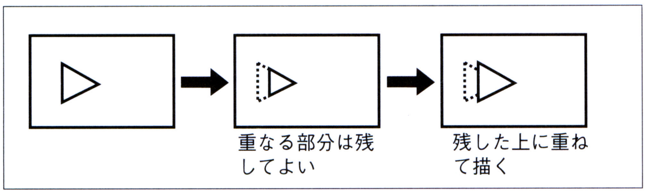
① **キャラクタを描きなおしていく（第4**



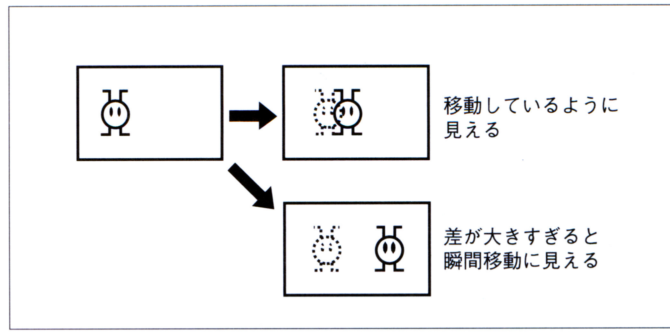
《第4図》キャラクタを描きなおしていく方法



《第5図》全画像を消してから表示しなおす方法



《第6図》違いを修正していく方法



《第7図》位置の変化量が大きいと瞬間移動に見える

図)。

表示していたキャラクタをいったん消去し，新しい場所に表示し直すというものです。プログラム的には簡単になりますが，キャラクタが重なっている場合などは他のキャラクタが消されてしまったりする場合があります。

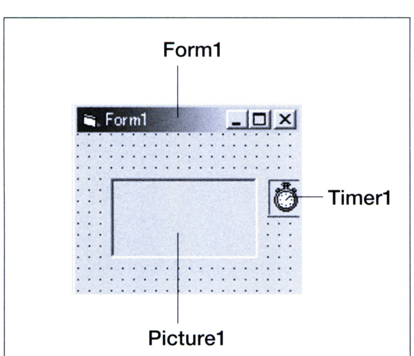
② 画像を描きなおして表示していく（第5図）。

ゲームフィールド全体をいったん消去し，何も無い状況にしてからすべてのキャラクタを表示するというものです。表示は正確で，キャラクタの重なりなどに対する注意もあまり必要ありませんが，すべてのキャラクタを表示させる必要があるため処理能力が必要になります。

③ 違いを順次修正していく（第6図）。

古い表示内容と新しい表示内容を比較して，違いがある部分だけ表示し直すというものです。画面に対する操作は少なくなりますが，どの部分が変化していてどの部分がそのままであるかという点を判断するのが難しくなり，プログラムが複雑になります。

市販ゲームソフトなどでは，ほとんどのものが②の方式を用いています。市販ゲームソフトはCコンパイラやアセンブラなどでプログラムが記述されているため，処理能力に余裕があり，②のような豪快な方式が使えます。



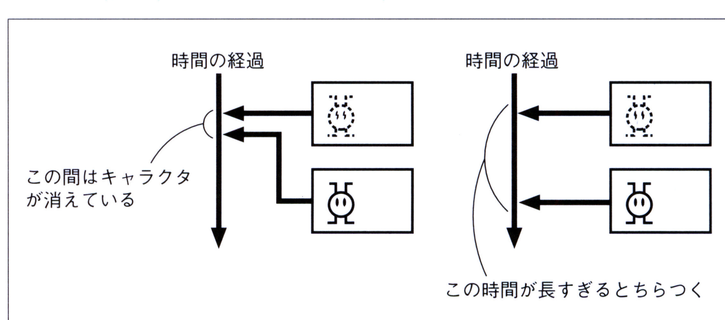
フォームの配置図

Visual BASICでのキャラクタ移動表現

Visual BASICはPコードインタプリタであるため，処理能力の点では不十分です。これは，コンパイラとインタプリタを比較すると桁違いの能力差がでてしまうためです。また，Visual BASIC自体，あまりゲームプログラミングを考慮していないため，②の手法で必要となるバックグラウンド画面を効果的に利用するのが難しくなっています（DirectXを利用すれば可能）。そこで，ここではキャラクタを描きなおしていく方法を使うことにします。

Visual BASICでグラフィックを表示する方法としては，ピクチャボックスを用意して，そこに描画指令を出すのが一般的です。この描画命令を利用することでキャラクタの表示を行いますが，背景色で描画することでキャラクタを消去することもできます。

そこで，キャラクタを描画するサブルーチンを用意し，このサブルーチンを呼び出す際に，キャラクタを描画するのか消去するのかをカラーコードの指定によって指示するようにします。キャラクタを表示する際には背景色以外のカラーコードを指定し，消去する際には背景色を指定するわけです。このようなサブルーチンをキャラクタごとに用意し，そのサブルーチンを呼び出してキャラクタの消去と描画を行います。



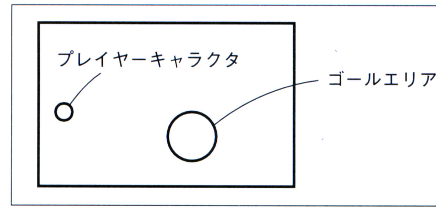
《第8図》消えている時間が長いとちらつく

アニメーション効果によってキャラクタの移動を表現するためには，元の位置に表示されているキャラクタを消去し，新しい位置にキャラクタを描画し直します。ここで注意しなければならないのは，新しい位置と元の位置の違いと，キャラクタが表示されていない時間の長さです。位置の変化が大きすぎると，キャラクタが瞬間移動したかのように見えてしまい，通常の移動を行ったようには見えません（**第7図**）。よって，キャラクタの移動量を大きくしすぎないように注意します。

また，キャラクタを消去してから新しい場所に描画するまでの時間が長くなると，キャラクタが消えている時間が長くなり，ちらついて見えるようになってしまいます。そこで，キャラクタの移動処理を行う部分と，キャラクタの描き直しを行う部分を分離し，消去したらすぐに新しい場所に描画できるようにします（**第8図**）。これによって，キャラクタが消えている時間を最小限に抑えることができます。



《写真1》「ドンピシャムーブゲーム」のプレイ画面。
ゴールした時の得点は最大100点。ゴールエリア中心付近で停止しないと100点にはならず，ゴールエリアの縁でゴールしてしまうと得点はかなり少なくなる。よって，間違ってゴールエリアの縁で停止してしまってゴールとなってしまう「ミスゴール」が無いように注意しなければならない。なお，ゴールエリアは徐々に小さくなっていくため，後半のゴールの方はミスゴールが発生しにくくなり，高得点を狙いやすい。



《第9図》ドンピシャムーブゲームの画面構成

ドンピシャムーブゲーム

それでは，実際にVisual BASICで動きを表現したプログラムを実行してみましょう。プログラムは**リスト1**です。

リスト1を実行すると，画面には**第9図**のようなゲームフィールドが表示されます。中央にはプレイヤーが操作するプレイヤーキャラクタが白く表示されます。このプレイヤーキャラクタを[2]，[4]，[6]，[8]のキーで上下左右に加速し，ゲームフィールド上に青く表示されているゴールまで移動させてください。ただし，ゴールエリア内で停止（速度をある一定速度より落とす）しないとゴールしたことにはなりません。ゴールするとその瞬間のゴール中心点までの距離に応じた得点が加算され，次のゴールが表示されます。なるべくゴール中心点で停止する方が高得点になります。制限時間内になるべく多くのゴールをクリアしてください。

リスト1ではプレイヤーキャラクタを加速がつくように移動させ，ゴールした場合はゴールマークを移動させる必要があります。プレイヤーキャラクタの座標はplayerxとplayeryに格納されています。キー入力に応じてこの座標を変化させてやれば，プレイヤーキャラクタの移動は行えますが，キー入力に応じて即座に移動方向が変化してしまい，加速がつくような移動にはなりません。そこで，プレイヤーキャラクタの速度として，playervxとplayervyを用意し，この速度をキー入力に応じて変化させてやるようにします。こうすると，キー入力に応じて即座に変わるのは速度になり，急に移

《リスト1》今月のサンプル『ドンピシャムーブゲーム』

```
Option Explicit

Dim keystat%(9) 'テンキーの状態を記録する変数
Dim keytrig0%, keytrig1%, keytrig2%, keytrig3% 'トリガー用キーの状態を記録する
変数
Dim playerx!, playery! 'プレイヤーキャラクタの座標
Dim playervx!, playervy! 'プレイヤーキャラクタの速度
Dim goalx!, goaly! 'ゴールエリアの座標
Dim goalsize! 'ゴールエリアの半径
Dim hiscore%, score% 'ハイスコアとスコア
Dim ti% '残り時間
Dim dt! '刻み時間
Dim gamestat% 'ゲーム進行状況

Private Sub Form_Load()
    hiscore = 0
    gameinit
End Sub

Private Sub gameinit() 'ゲーム初期化
    Dim ox%, oy%, x%, y%, i%
    With Form1
        .Width = Screen.TwipsPerPixelX * 640 'フォームのサイズを設定
        .Height = Screen.TwipsPerPixelY * 480
        .Caption = "ドンピシャMoveGame"
    End With
    With Picture1
        .Width = Screen.TwipsPerPixelX * 576 'ピクチャボックスのサイズを設定
        .Left = Screen.TwipsPerPixelY * 32
        .Height = Screen.TwipsPerPixelY * 432
        .Top = 0
        .ScaleMode = 3
        .ScaleHeight = 432
        .ScaleWidth = 576
        .BackColor = RGB(0, 0, 0)
        .ForeColor = RGB(255, 255, 255)
        .FillStyle = 0
        .Font = "ＭＳ ゴシック"
        .FontSize = 8
        .FillStyle = vbFSTransparent
    End With
    Picture1.Cls
    Form1.Show

    With Picture1
        .CurrentX = 0
        .CurrentY = 0
        .FontSize = 10
        .ForeColor = RGB(255, 255, 255)
    End With
    Picture1.Print "High Score " & hiscore

    playerx = 10 'プレイヤー初期座標
    playery = 216
    playervx = 0 'プレイヤー初期速度
    playervy = 0

    goalsize = 60
    setgoalposition 'ゴール座標設定
    putplayer playerx, playery, RGB(255, 255, 255)
    putgoal goalx, goaly, RGB(0, 0, 255)

    score = 0
    putscore
    ti = 3000
    puttime

    dt = 0.1 '刻み時間
    gamestat = 0 'ゲーム進行フラグ初期化
    Timer1.Interval = 20 'インターバルタイマ起動
End Sub

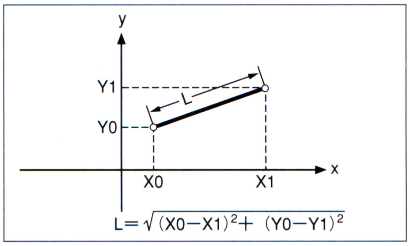
Private Sub Picture1_MouseUp(button%, shift%, x!, y!)
    If (button And 2) = 2 Then 'マウス右ボタンクリックで強制終了
        End
    End If
End Sub

Private Sub Picture1_KeyDown(key%, shift%)
    If key = 27 Then 'ESCキーで強制終了
        End
    End If
    If key > 96 And key < 106 Then 'テンキー設定
        keystat(key - 96) = 1
    End If
    If key = 32 Then keytrig0 = 1 'トリガキー設定
    If key = 90 Then keytrig1 = 1
    If key = 88 Then keytrig2 = 1
    If key = 13 Then keytrig3 = 1
End Sub

Private Sub Picture1_KeyUp(key%, shift%)
    If key > 96 And key < 106 Then
        keystat(key - 96) = 0
    End If
    If key = 32 Then keytrig0 = 0
    If key = 90 Then keytrig1 = 0
    If key = 88 Then keytrig2 = 0
    If key = 13 Then keytrig3 = 0
End Sub

Private Sub Timer1_Timer()
    Dim newplayervx!, newplayervy!, newplayerx!, newplayery!
    Dim l!
    If gamestat = 0 Then 'ゲーム開始前
        If keystat(2) = 1 Or keystat(4) = 1 Or keystat(6) = 1 Or
keystat(8) = 1 Then gamestat = 1 'キーを押すまで待機
    End If
    If gamestat = 1 Then 'ゲーム中
        'プレイヤーキャラクタの処理
```

つぎのページに続く



《第10図》2点間の距離

動方向が変化することはなくなって加速がつくような移動になります。

移動を表現するためには，元々表示されていたキャラクタを消去し，新しい場所に表示し直す必要があります。さらに，キャラクタのちらつきを避けるためには，消去してから表示し直すまでの時間を最小限にする必要があります。このため，あらかじめ新しいキャラクタの座標newplayerxとnewplayeryを計算しておき，消去したらすぐに表示し直せるようにしています。

プレイヤーキャラクタがゴールしたかどうかの判断は，キャラクタがゴールエリア範囲内に存在しているかどうかという点と，キャラクタの速度が十分小さいかどうかという点をチェックする必要があります。範囲のチェックは，キャラクタ座標とゴール座標の距離を計算してチェックしています。座標（X0，Y0）と（X1，Y1）の間の距離Lは，**第10図**のような式で計算することができます。同様に，速度が十分小さいかどうかについても，速度の値を座標と見なして**第10図**に当てはめて計算し，チェックしています。

ゴールした後はゴールエリアを新しい場所に設定しなければなりません。あまりにもプレイヤーキャラクタに近い場所になってしまうと面白くありませんから，ある程度遠くに設定する必要があります。ゴールエリアの座標を設定するsetgoalposition関数では，まず乱数でゴールエリアの座標を計算し，プレイヤーキャラクタとの距離が十分離れている場合にのみ，その座標に設定し，そうでない場合は再び乱数でゴールエリアの座標を計算させるようにしています。これによって，常にゴールエリアは，ある程度離れた場所に移動するようになります。

おわりに

キャラクタの移動表現は，ゲームの基本的な表現になります。Visual BASICの場合は，今回のリストのように，消してから表示し直す方法が比較的簡単です。キャラクタが消えている時間を最小限に抑えれば，ちらつきもそれほど気にならなくなります。

次回は，物の速度を計算するプログラムを作ってみます。

《第1表》リスト1の主要変数一覧

playerx	プレイヤーキャラクタのＸ座標
playery	プレイヤーキャラクタのＹ座標
playervx	プレイヤーキャラクタのＸ方向の速度
playervy	プレイヤーキャラクタのＹ方向の速度
goalx	ゴールエリアのＸ座標
goaly	ゴールエリアのＹ座標
goalsize	ゴールエリアの半径
hiscore	ハイスコア
score	スコア
ti	残り時間
dt	移動量計算時に使う刻み時間
gamestat	ゲーム進行状況 0：ゲーム開始直前状態 1：ゲーム中 2：ゲームオーバー

```
    'プレイヤーキャラクタの処理
    '
    newplayervx = playervx
    newplayervy = playervy
    'キー状況に応じて速度を変更
    If keystat(2) = 1 Then newplayervy = newplayervy + 3
    If keystat(8) = 1 Then newplayervy = newplayervy - 3
    If keystat(4) = 1 Then newplayervx = newplayervx - 3
    If keystat(6) = 1 Then newplayervx = newplayervx + 3
    '速度リミッタ
    If Abs(newplayervx) > 50 Then newplayervx = playervx
    If Abs(newplayervy) > 50 Then newplayervy = playervy
    '速度更新
    playervx = newplayervx
    playervy = newplayervy
    '移動先の座標を計算
    newplayerx = playerx + playervx * dt
    newplayery = playery + playervy * dt
    '移動範囲制限
    If newplayerx < 10 Then newplayerx = 10
    If newplayerx > 566 Then newplayerx = 566
    If newplayery < 30 Then newplayery = 30
    If newplayery > 422 Then newplayery = 422
    putplayer playerx, playery, RGB(0, 0, 0) '元の場所のキャラクタを消去
    putplayer newplayerx, newplayery, RGB(255, 255, 255) '新しい位置にキャラクタを描画
    '座標更新
    playerx = newplayerx
    playery = newplayery
    '
    'ゴールエリアの処理
    '
    putgoal goalx, goaly, RGB(0, 0, 255) '表示し直し
    'プレイヤーキャラクタがゴールエリアに入っているかどうかのチェック
    l = Sqr((playerx - goalx) ^ 2 + (playery - goaly) ^ 2)
    If l < goalsize - 10 Then
        'プレイヤーキャラクタの速度が十分小さくなっているかどうかのチェック
        If Sqr(playervx ^ 2 + playervy ^ 2) < 5 Then
            If l < 10 Then l = 10
            score = score + 1000 / l '得点増加
            putscore
            putgoal goalx, goaly, 0 'ゴールを消去
            setgoalposition '新しい位置を計算
            If goalsize > 20 Then goalsize = goalsize - 2 'だんだんゴールを小さくする
        End If
    End If
    '
    '時間経過
    '
    ti = ti - 1
    If ti < 1 Then gamestat = 2 '時間が無くなったらゲーム終了
    puttime
End If
If gamestat = 2 Then 'ゲームオーバー状態
    With Picture1
        .CurrentX = 230
        .CurrentY = 200
        .FontSize = 15
        .ForeColor = RGB(Rnd * 255, Rnd * 255, Rnd * 255)
    End With
    Picture1.Print "Game Over"
    If score > hiscore Then hiscore = score
    putscore
    If keytrig3 = 1 Then
        gameinit
    End If
  End If
End Sub

Private Sub putscore() 'スコアの表示
  Picture1.Line (270, 0)-(375, 15), 0, BF
  With Picture1
    .CurrentX = 270
    .CurrentY = 0
    .FontSize = 10
    .ForeColor = RGB(255, 255, 255)
  End With
  Picture1.Print "Score " & score
End Sub

Private Sub puttime() '残り時間の表示
  Picture1.Line (470, 0)-(575, 15), 0, BF
  With Picture1
    .CurrentX = 470
    .CurrentY = 0
    .FontSize = 10
    .ForeColor = RGB(255, 255, 255)
  End With
  Picture1.Print "Time " & ti
End Sub

Private Sub setgoalposition() 'ゴールエリアの座標設定
  Dim f, l
  f = 0
  While f = 0
    goalx = Rnd(1) * 400 + 88
    goaly = Rnd(1) * 300 + 66
    l = Sqr((playerx - goalx) ^ 2 + (playery - goaly) ^ 2)
    If l > 100 + goalsize Then f = 1
  Wend
End Sub

Private Sub putplayer(x, y, c) 'プレイヤーキャラクタの表示
  Picture1.Circle (x, y), 10, c
End Sub

Private Sub putgoal(x, y, c) 'ゴールエリアの表示
  Picture1.Circle (x, y), goalsize, c
End Sub
```



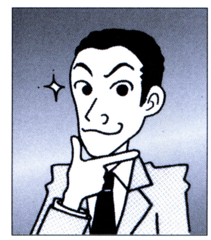
はじめよう Microsoft ExcelでBASICプログラミング

協力 富士通南九州システムエンジニアリング

第12回 コンピュータ対戦型のパズルゲームを作ろう(5)

この連載では，Microsoft Excelを使ってBASICのプログラミングを行います。今回はミニ・マックス法を使用して，前回作成したオセロ・ゲームを完成させます。

TEXT & PROGRAMING　**椛島たけし**



最近W杯をテーマにした商品やCFが増えている。サッカーと関係の無い商品を、強引に結び付けているモノもあるが結構楽しめる。
まさか、W杯モデルのPCが発売されたりはしないよね。

ミニ・マックス法とα-β法

先月号では先読みの方法として，ミニ・マックス法とα-β法を紹介しました。簡単に復習しておきましょう。

ミニ・マックス法

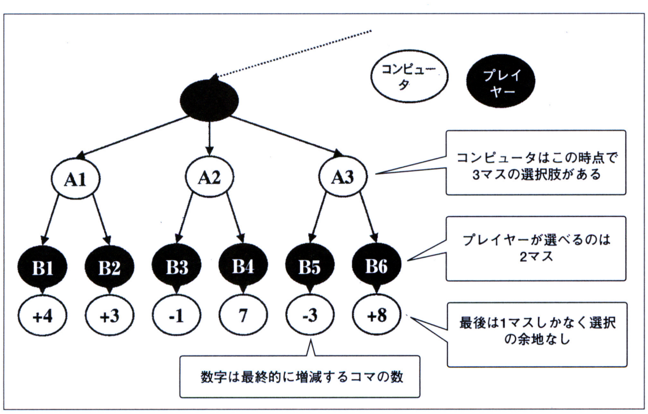
自分の手に対し，相手は常に最良の手，つまり自分にとって最悪の手を常に選択すると想定して，手を決定する方法です(第1，2図)。すべての手について調べるので，あまり先まで読むと処理時間がかかります。

α-β法

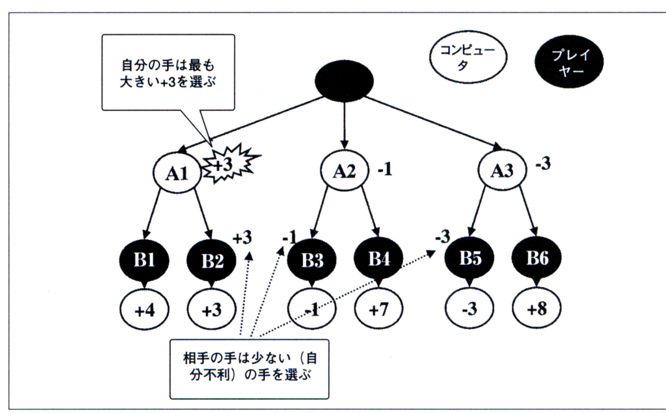
ミニ・マックス法の応用編です。先読みをする必要が無いと思われる部分をカットし，先読みの数を減らすのが特長です(第3図)。最後の手まで先読みを行わない場合，いわゆる大逆転の手を見逃すことがあります。

配列とユーザー定義型

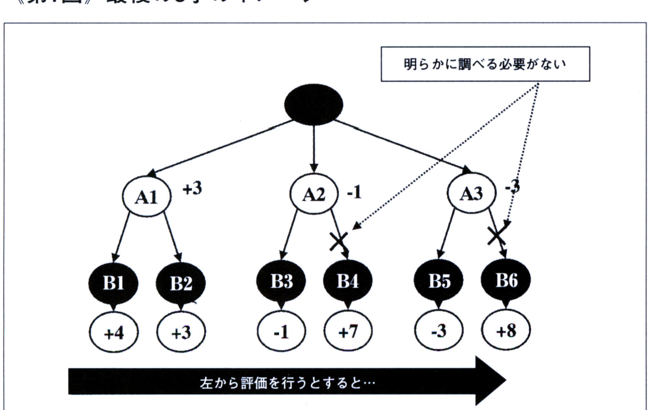
今回は，ミニ・マックス法を利用して，手の先読みを行うプログラムを作成しま



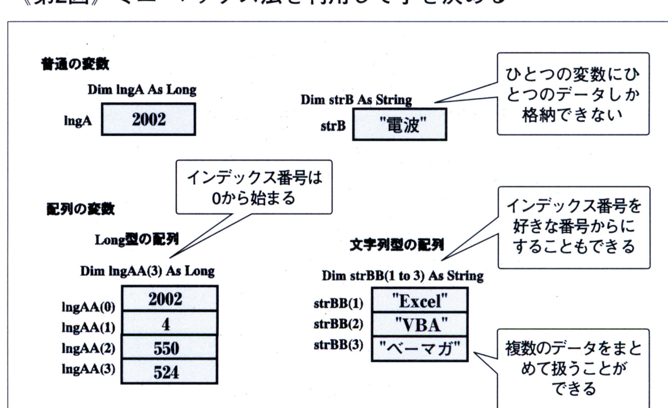
《第1図》最後の3手のイメージ



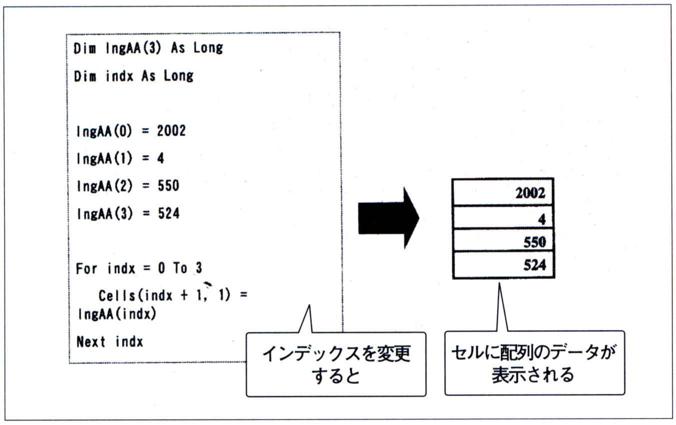
《第2図》ミニ・マックス法を利用して手を決める



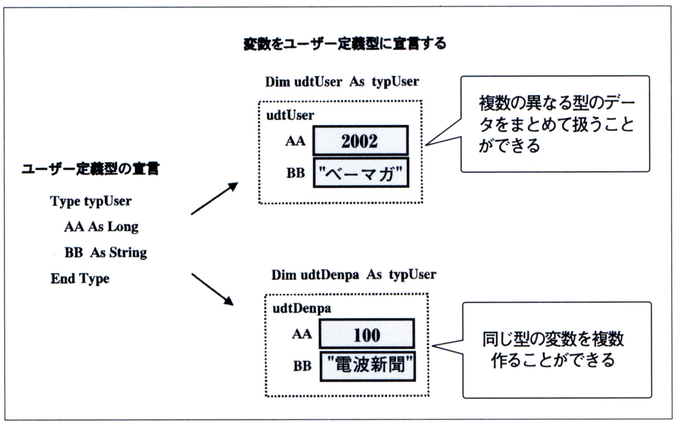
《第3図》α-β法を利用して手を評価すると効率的になる



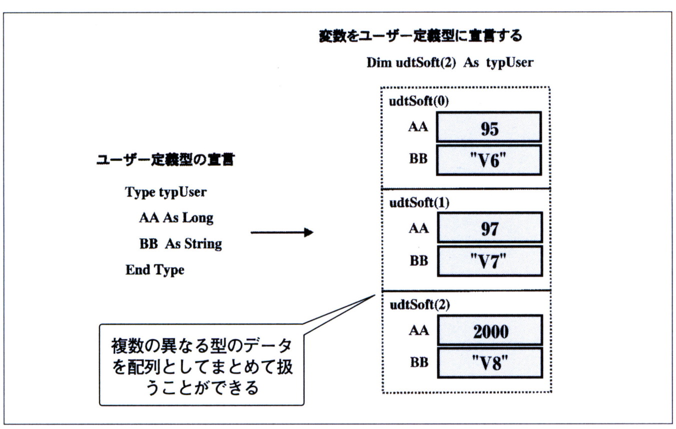
《第4図》配列とは？



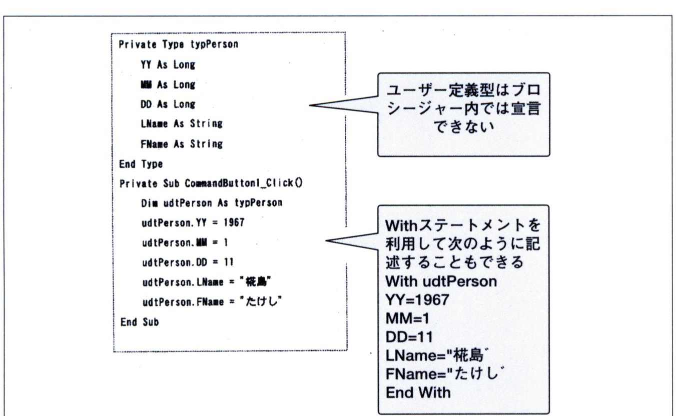
《第5図》配列の利用例



《第6図》ユーザー定義型とは？



《第7図》ユーザー定義型を配列することもできる



《第8図》ユーザー定義型の例

すが，盤面や手の状態を格納するために，配列とユーザー定義型を多用する必要があります。そこで，配列とユーザー定義型について説明しておくことにします。

配列

配列とは，同じデータ型の複数のデータ(要素)を一組にまとめた変数のことです。変数名は一つですが，インデックス番号(添字)を指定することで，複数のデータを扱うことができます(**第4，5図**)。

ユーザー定義型

ユーザー定義型とは，異なるデータ型等を複数組み合わせて，独自のデータ型としたものです。ユーザー定義型の要素それぞれに要素名を付け，データを区別します。C言語では構造体と呼ばれます。また，要素名をメンバと呼ぶ場合もあります(**第7，8図**)。

ユーザー定義型配列

ユーザー定義型を配列にしたものです。ユーザー定義型自体がデータ型ですので，データの形式があるユーザー定義型の配

《リスト1》いろいろなデータをプロシージャーに渡す例

```
'ユーザー定義型の宣言
Private Type typPerson
    No  As Long
    YY As Long
    MM As Long
    DD As Long
    LName As String
    FName As String
End Type

Private Sub CommandButton1_Click()
    Dim lngSal(2) As Long 'Long型の配列宣言
    Dim udtPerson As typPerson 'typPerson型の変数宣言
    Dim udtEmp(9) As typPerson 'typPerson型の配列
    Dim ctr As Long 'カウンタ用の変数宣言

    '配列にデータを代入
    lngSal(0) = 1000
    lngSal(1) = 2000
    lngSal(2) = 3000

    'ユーザー定義型の変数にデータを代入
    With udtPerson
        .YY = 1967
        .MM = 1
        .DD = 11
        .LName = "椛島"
        .FName = "たけし"
    End With

    'ユーザー定義型の配列にデータを代入
    For ctr = 0 To 9
        udtEmp(ctr).No = ctr + 100
    Next ctr

    'サブプロシージャーを呼び出す
    Call procParamTest(lngSal(), udtPerson, udtEmp())
End Sub

'変数を受け取るサブプロシージャ
Private Sub procParamTest(DimData() As Long, _
                    TypeData As typPerson, _
                    TypeDimData() As typPerson)
    Dim ctr As Long 'カウンタ用の変数

    'セルにユーザー定義型の変数のデータを入力
    Cells(1, 1) = TypeData.LName
    Cells(2, 1) = TypeData.FName

    'セルに配列のデータを入力
    For ctr = 0 To 2
        Cells(ctr + 1, 2) = DimData(ctr)
    Next ctr

    'セルにユーザー定義型の配列のデータを入力
    For ctr = 0 To 9
        Cells(ctr + 1, 3) = TypeDimData(ctr).No
    Next ctr

End Sub
```

列変数を作成することが可能です。

変数をプロシージャに引き渡す

プロシージャーに変数を渡す場合，引数に指定することで引き渡すことができます。もちろん，配列自体をプロシージャーに渡すことも可能です。簡単なプログラムを作成してテストしてみると使い方がよくわかるはずです(**リスト1**)。

再帰プロシージャー

ミニ・マックス法をプログラミングする場合，どのぐらい先まで手を読むかがポイントとなります。そのとき，2手先まで読む処理と，3手先を読む処理が別々ではプログラムリストが複雑で長くなってしまいます。次の手を読む処理自体は，共通に使用できますので，上手くまとめるとシンプルなプログラムにすることができます。

回数が決まっていない処理の場合，通常はループ回数を変数することで解決できます。しかし，ミニ・マックス法で先読みを行う場合，単純にループ処理を利用するだけでは解決しません。したがって，このような処理には再帰プロシージャーを作成するのが非常に便利です(**第9図**)。

再帰プロシージャーとは，自分自身を呼び出すプロシージャーのことです。再帰プロシージャーは，抜け出すきっかけが必要です。ループ処理の場合は，ループを抜ける条件を指定しますが，再帰プロシージャーでは，自分自身を呼び出さない条件を指定することで，再帰の回数を決めることができます。

慣れないうちは，再帰プロシージャーの動作を頭の中で考えると，こんがらがってしまいそうですね。しかし使っているうちに，応用範囲が広い便利な方法だとわかると思います(**第10図**)。

2手先まで読んでみる

ミニ・マックス法を利用して，2手先まで読むプログラムを作成してみます。わかりやすくするために，ここでは，再帰プロシージャーは使用しないことにしましょう。

実際に手の先を読む場合，盤面にコマを並べて，「ここに打ったとして」「ここに打たれたら」と予測していきますね。まずはこの方法を流れ図にしてみます(**第11図**)。

2手先，つまりコンピュータの手と，それに対するプレイヤーの手を調べる場合それほど複雑な処理ではありませんね。

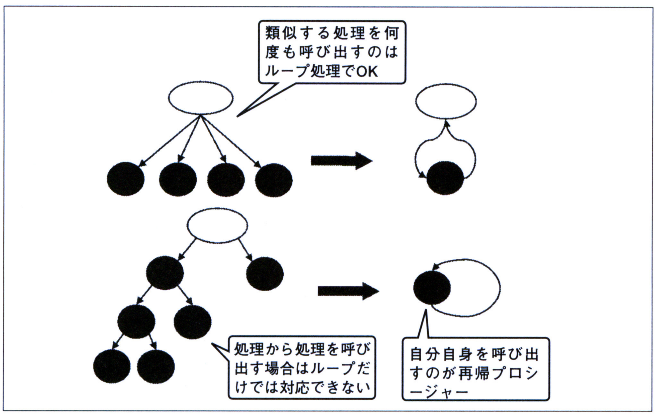
プログラミングのコツ

盤面の状態を保存した配列がありますが，手が進むとその分だけ配列が必要となります。最初から配列を用意しておくと大変ですね。そのため，現在の盤面をプロシージャーに渡し，プロシージャーで盤面をコピーすることにします。そうすれば，プロシージャーは初めから多くの配列を用意しておく必要はありません。

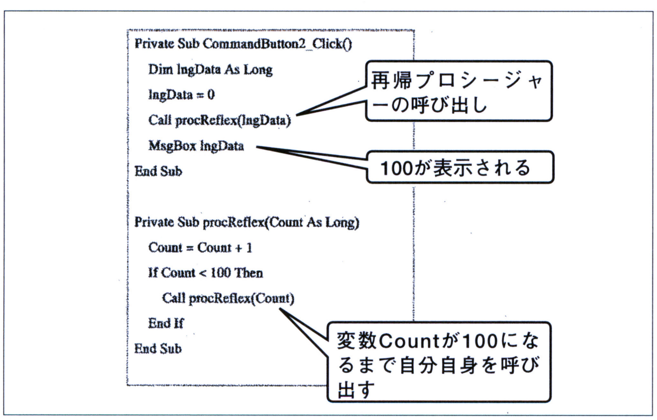
また，いわゆる手の保存にはユーザー定義型を利用します。ユーザー定義型には手を表す行と列，その手でとれるコマの数，ミニ・マックス法で得られるポイントを要素として定義しておきます。

2手先まで完成

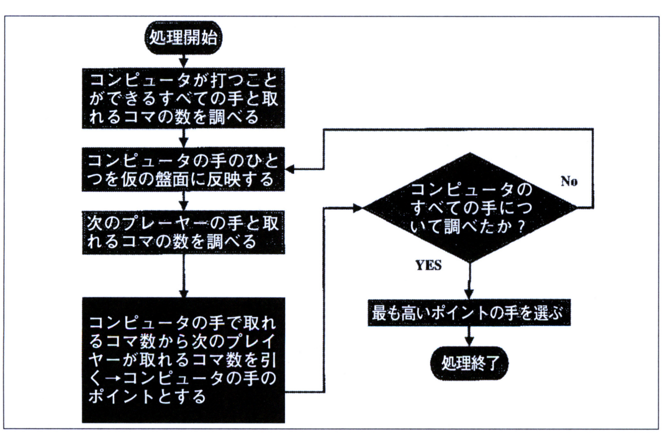
さて，2手先まで読むプログラム・リ



《第9図》再帰プロシージャーとは？



《第10図》簡単な再帰プロシージャーの例



《第11図》ミニ・マックス法で2手先まで読む流れ図

《リスト2》2手先まで読むプログラム・ソフト

```
Option Explicit
Private Const PIECE_CHR = "●" 'コマを表す文字
Private Const COL_BLACK = &H0       'コマの色(黒)
Private Const COL_WHITE = &HFFFFF   'コマの色(白)
Private Const VAL_BLACK = -1    '黒の値
Private Const VAL_WHITE = 1     '白の値
Private Const VAL_SPACE = 0     '空の値

Private Const BOARD_TOP = 1     '盤の上位置
Private Const BOARD_LEFT = 1    '盤の左位置
Private Const BOARD_SIZE = 8    '盤のサイズ

Private Const ERR_OUTOF = 2&    'エラー(盤外)
Private Const ERR_SAME = 4&     'エラー(同じマス)
```

つぎのページに続く

```
Private Const ERR_MISS = 8&     'エラー(打てないマス)
'序盤ポイント表用データ
Private Const POINTTABLE_DATA1 = "82433428" & _
                                 "21555512" & _
                                 "45655654" & _
                                 "35500553" & _
                                 "35500553" & _
                                 "45655654" & _
                                 "21555512" & _
                                 "82433428"

'中盤ポイント表用データ
Private Const POINTTABLE_DATA2 = "82766728" & _
                                 "21344312" & _
                                 "73455437" & _
                                 "64500546" & _
                                 "64500546" & _
                                 "73455437" & _
                                 "21344312" & _
                                 "82766728"

'終盤ポイント表用データ
Private Const POINTTABLE_DATA3 = "85777758" & _
                                 "54666645" & _
                                 "76666667" & _
                                 "76600667" & _
                                 "76600667" & _
                                 "76666667" & _
                                 "54666645" & _
                                 "85777758"


'コンピューターの手の格納用のユーザ定義型
Private Type typIdea
   Row As Long
   Col As Long
   Rev As Long
   Pnt As Long
End Type

Dim BoardStatus(0 To 9, 0 To 9) As Long 'オセロ盤の状況保存用配列
Dim plngCompColor As Long  'コンピューターのコマの色
Dim plngPlayerColor As Long 'プレイヤーのコマの色
Dim udtIdea(60) As typIdea  '手格納用ユーザ定義配列
Dim plngTurnCount As Long   'ターン数カウント用配列
Dim PointTable1(1 To 8, 1 To 8) As Long 'ポイント表用配列(序盤)
Dim PointTable2(1 To 8, 1 To 8) As Long 'ポイント表用配列(中盤)
Dim PointTable3(1 To 8, 1 To 8) As Long 'ポイント表用配列(終盤)

'スタートボタンが押された時の処理
Private Sub cmdStart_Click()
   Dim lngRowCtr As Long
   Dim lngColCtr As Long

   plngPlayerColor = VAL_BLACK
   plngCompColor = VAL_WHITE
   plngTurnCount = 1
   '状態配列をクリアする
   For lngRowCtr = 1 To 8
      For lngColCtr = 1 To 8
         BoardStatus(lngRowCtr, lngColCtr) = 0
      Next lngColCtr
   Next lngRowCtr

   Call procSetPoinTable
   '初期のコマの配置を行う
   BoardStatus(4, 4) = VAL_WHITE
   BoardStatus(5, 5) = VAL_WHITE
   BoardStatus(4, 5) = VAL_BLACK
   BoardStatus(5, 4) = VAL_BLACK

   '画面表示処理を呼び出す
   Call procDrawPiece

   '一旦盤の外を選択する
   Cells(BOARD_TOP + BOARD_SIZE, BOARD_LEFT +
BOARD_SIZE).Activate
End Sub

'ワークシートの選択セルが変更された時
'≒マウスがクリックされたとき
Private Sub Worksheet_SelectionChange(ByVal Target As Range)
   Dim lngClickRow As Long 'クリックされた行
```

```
   Dim lngClickCol As Long 'クリックされた列
   Dim lngRetCode As Long  'プロシージャの戻り値
   Dim boolComp As Boolean
   Dim boolPlayer As Boolean

   Dim lngPlayer As Long  'プロシージャの戻り値
   Dim lngComp As Long  'プロシージャの戻り値

   Dim lngPlayerPoint As Long
   Dim lngCompPoint As Long
   Dim strResultMsg As String

   If plngTurnCount > 60 Then Exit Sub

   'プレイヤーのコマのチェック処理の呼び出し
   lngRetCode = procPlayerCheck(Target(1, 1).Row, Target(1, 1).Column)
   If lngRetCode = -1 Then  '戻り値が-1(正しいマスを選択した場合)
      BoardStatus(Target(1, 1).Row, Target(1, 1).Column) = -1
      Call procDrawPiece  'コマを表示する処理

      'プレイヤーが打つ手がある間ループ
      Do Until lngPlayer > 0
         'コンピューターの打てる場所のチェック
         lngComp = procIdeaCheck(BoardStatus(), plngCompColor,
udtIdea())
         If lngComp > 0 Then  '打てる場所がある場合
            'コンピュータの手を決める処理
            Call procCompIdea(lngComp)
         End If
         'プレイヤーに打つ手があるかチェック
         lngPlayer = procIdeaCheck(BoardStatus(), plngPlayerColor,
udtIdea)

         If lngPlayer = 0 And lngComp = 0 Then
            '得点計算処理
            Call procPointCalc(lngPlayerPoint, lngCompPoint)
            strResultMsg = "終了です。" & vbCrLf & _
                           "プレイヤー " & CStr(lngPlayerPoint) &
vbCrLf & _
                           "コンピューター " & CStr(lngCompPoint)
            MsgBox strResultMsg
            Exit Do
         ElseIf lngPlayer = 0 Then
            MsgBox "プレイヤーは打つ手がありません。コンピューターの番
です。"
         ElseIf lngComp = 0 Then
            MsgBox "コンピューターは打つ手がありません。あなたの番です。
"
         Else
         End If
      Loop

   End If

End Sub


'クリックされたマスについて調べる処理
Private Function procPlayerCheck(lngRow As Long, lngCol As Long) As Long
   Dim lngRowCtr As Long
   Dim lngColCtr As Long
   Dim lngReverse As Long
   Dim lngRetRev As Long
   Dim lngRetCode As Long

   'マス範囲外のチェック
   procPlayerCheck = 0
   Select Case lngRow
   Case BOARD_TOP To BOARD_TOP + BOARD_SIZE - 1
      Select Case lngCol
      Case BOARD_LEFT To BOARD_LEFT + BOARD_SIZE - 1
      'すでにコマが無いかチェック
         If BoardStatus(lngRow, lngCol) <> VAL_SPACE Then
            procPlayerCheck = ERR_SAME
         Else
            procPlayerCheck = -1
         End If
      Case Else
         procPlayerCheck = ERR_OUTOF
      End Select
   Case Else
      procPlayerCheck = ERR_OUTOF
   End Select
```

→ つぎのページに続く

```
    If procPlayerCheck <> -1 Then Exit Function

    lngRetCode = procReverseCheck(BoardStatus(), plngPlayerColor, lngRow,
lngCol)
    If lngRetCode > 0 Then
        Call procReverse(BoardStatus(), plngPlayerColor, lngRow, lngCol)
        procPlayerCheck = -1
    Else
        procPlayerCheck = ERR_MISS
    End If


End Function

'打つ手があるか調べる
Private Function procIdeaCheck(BoardStatusTemp() As Long, CheckColor As
Long, udtIdeaTemp() As typIdea) As Long
    Dim lngRowCtr As Long '行カウンタ
    Dim lngColCtr As Long '列カウンタ
    Dim lngIdeaCtr As Long '手数のカウンタ
    Dim lngCompIdea As Long
    Dim lngRetCode As Long
    'コマを置ける位置の確認
    For lngRowCtr = 1 To 8
        For lngColCtr = 1 To 8
            '空白のマスの場合
            If BoardStatusTemp(lngRowCtr, lngColCtr) = VAL_SPACE Then
                'コマを返せるか調べる
                lngRetCode = procReverseCheck(BoardStatusTemp(),
CheckColor, lngRowCtr, lngColCtr)
                If lngRetCode > 0 Then
                    lngIdeaCtr = lngIdeaCtr + 1
                    With udtIdeaTemp(lngIdeaCtr)
                        .Row = lngRowCtr
                        .Col = lngColCtr
                        .Rev = lngRetCode
                    End With
                End If
            End If
        Next lngColCtr
    Next lngRowCtr

    procIdeaCheck = lngIdeaCtr

End Function
'コンピュータの手を決める処理
Private Sub procCompIdea(IdeaCount As Long)
    Dim lngCompIdea As Long 'コンピューターが打てる手の場所数
    Dim lngWorkCtr As Long 'ワークカウンタ
    Dim lngMuchCtr As Long '高ポイント
    Dim lngMuchIdea(64) As Long '高ポイント格納用配列
    Dim lngMaxValue As Long '最高ポイント
    Dim lngPointValue As Long   'マスのポイント
    Dim lngNextCount As Long

    Application.Cursor = xlWait 'カーソルを砂時計にする

    lngMaxValue = 0
    lngMuchCtr = 1
    lngMuchIdea(1) = 0

    lngNextCount = 60 - plngTurnCount

    lngCompIdea = procMinMax(BoardStatus(), _
                            plngCompColor, _
                            IdeaCount, _
                            udtIdea(), _
                            lngNextCount)
   'コマを返す処理
    Call procReverse(BoardStatus(), _
                    plngCompColor, _
                    udtIdea(lngCompIdea).Row, _
                    udtIdea(lngCompIdea).Col)
    Call procDrawPiece '盤面を描く
    Application.Cursor = xlDefault 'カーソルを戻す

End Sub


'得点計算処理
Private Sub procPointCalc(PlayerPoint As Long, CompPoint As Long)
    Dim lngRowCtr As Long
    Dim lngColCtr As Long
    plngPlayerColor = VAL_BLACK
    plngCompColor = VAL_WHITE

    '状態配列をクリアする
    For lngRowCtr = 1 To 8
        For lngColCtr = 1 To 8
            Select Case BoardStatus(lngRowCtr, lngColCtr)
            Case plngPlayerColor
                PlayerPoint = PlayerPoint + 1
            Case plngCompColor
                CompPoint = CompPoint + 1
            End Select
        Next lngColCtr
    Next lngRowCtr

End Sub

'ポイント表を読み込む処理
Private Sub procSetPoinTable()
    Dim lngRowCtr As Long  '行用カウンター
    Dim lngColCtr As Long  '列用カウンター
    Dim lngCellCtr As Long 'マスの位置

    For lngRowCtr = 1 To 8
        For lngColCtr = 1 To 8
           lngCellCtr = lngCellCtr + 1
           PointTable1(lngRowCtr, lngColCtr) = _
                    CLng(Mid(POINTTABLE_DATA1, lngCellCtr, 1))
           PointTable2(lngRowCtr, lngColCtr) = _
                    CLng(Mid(POINTTABLE_DATA2, lngCellCtr, 1))
           PointTable3(lngRowCtr, lngColCtr) = _
                    CLng(Mid(POINTTABLE_DATA3, lngCellCtr, 1))
        Next lngColCtr
    Next lngRowCtr
End Sub

'MinMax法で2手先まで読む
Private Function procMinMax(BoardStatusTemp() As Long, _
                            CheckColor As Long, _
                            IdeaCount As Long, _
                            udtIdeaNext() As typIdea, _
                            NextCount As Long) As Long

    Dim BoardStatusNext(9, 9) As Long  '手を打った後の盤面
    Dim udtIdeaNNext(64) As typIdea     '次ぎの手を格納するユーザ定義型
    Dim lngWorkCtr As Long             'ワークカウンタ
    Dim lngNNextCtr As Long            'ワークカウンタ(次の手)
    Dim lngMuchCtr As Long             '最高ポイント用カウンタ
    Dim lngMuchIdea(64) As Long          '高ポイント格納用配列
    Dim lngMaxValue As Long            '最高ポイント
    Dim lngNMaxValue As Long            '最高ポイント
    Dim lngMinValue As Long            '最低ポイント
    Dim lngPointValue As Long          'マスのポイント
    Dim lngNNextIdea As Long            '次ぎの手の数
    Dim lngNCheckColor As Long           '調べる色

    '変数の初期化
    lngMaxValue = -64
    lngMuchCtr = 1
    lngMuchIdea(1) = 0

    '調べる色を判別する
    If CheckColor = VAL_WHITE Then
        lngNCheckColor = VAL_BLACK
    Else
        lngNCheckColor = VAL_WHITE
    End If

    '次の手(Max)の数だけ調べる
    For lngWorkCtr = 1 To IdeaCount
        '盤面を複写する
        Call procCopyBoard(BoardStatusTemp(), BoardStatusNext())
        '次の手を盤面に反映する
        Call procReverseBoard(BoardStatusNext(), _
                    CheckColor, _
                    udtIdeaNext(lngWorkCtr).Row, _
                    udtIdeaNext(lngWorkCtr).Col)

        lngNNextIdea = -1 '変数の初期化
        '次ぎの次の手がいくつあるか調べる
        lngNNextIdea = procIdeaCheck(BoardStatusNext(), _
                            lngNCheckColor, udtIdeaNNext())
```

つぎのページに続く

```
        If lngNNextIdea = 0 Then '次の次の手が打てないなら
            udtIdeaNext(lngWorkCtr).Pnt = 64   '最も良い手とする
        Else
            lngNMaxValue = 0 '変数の初期化
            For lngNNextCtr = 1 To lngNNextIdea '次の次の手のすべてを調べる
                '最も多く取れる(取られる)次の次の手を調べる(Minを調べる)
                If lngNMaxValue < udtIdeaNNext(lngNNextCtr).Rev Then
                    lngNMaxValue = udtIdeaNNext(lngNNextCtr).Rev
                End If
                '次の手のポイントを次の次の手までのポイントとする(Max)
                udtIdeaNext(lngWorkCtr).Pnt = _
                                udtIdeaNext(lngWorkCtr).Rev - lngNMaxValue
            Next lngNNextCtr
        End If

        Select Case udtIdeaNext(lngWorkCtr).Pnt   '次の手(Max)のポイント
        Case Is = lngMaxValue '最大ポイントと同じ
            lngMuchCtr = lngMuchCtr + 1
            lngMuchIdea(lngMuchCtr) = lngWorkCtr '手の番号を格納
        Case Is > lngMaxValue '最大ポイントより大きい場合
            lngMuchCtr = 1
            lngMaxValue = udtIdeaNext(lngWorkCtr).Pnt '最大ポイントを保存
            lngMuchIdea(1) = lngWorkCtr '手の番号を格納
        Case Else
        End Select
    Next lngWorkCtr

    '最高ポイントの手が複数ある場合ランダムに手を選択する
    procMinMax = lngMuchIdea(Int((lngMuchCtr * Rnd) + 1))

End Function


'コマを表示する処理
Private Sub procDrawPiece()
    Dim lngRowCtr As Long '行カウンタ
    Dim lngColCtr As Long '列カウンタ

    '配列のすべてを調べる
    For lngColCtr = 1 To 8
        For lngRowCtr = 1 To 8
            Select Case BoardStatus(lngRowCtr, lngColCtr)
            Case VAL_SPACE  '空白なら
                Cells(lngRowCtr, lngColCtr).Value = ""
            Case VAL_BLACK  '黒なら
                'セルの文字色に黒を設定し、●を表示する
                Cells(lngRowCtr, lngColCtr).Font.Color = COL_BLACK
                Cells(lngRowCtr, lngColCtr).Value = PIECE_CHR
            Case VAL_WHITE  '白なら
                'セルの文字色に白を設定し、●を表示する
                Cells(lngRowCtr, lngColCtr).Font.Color = COL_WHITE
                Cells(lngRowCtr, lngColCtr).Value = PIECE_CHR
            End Select
        Next lngRowCtr
    Next lngColCtr

End Sub

'そのマスにコマを置けるか調べる
Private Function procReverseCheck(BoardStatusTemp() As Long,
CheckColor As Long, NowRow As Long, NowCol As Long) As Long
    Dim lngReverse As Long  '返すコマの総数
    Dim lngRetCode As Long  '返すコマの数

    lngReverse = 0
    lngRetCode = procCellsCheck(BoardStatusTemp(), CheckColor, NowRow,
NowCol, -1, -1) '左上
    lngReverse = lngReverse + lngRetCode
    lngRetCode = procCellsCheck(BoardStatusTemp(), CheckColor, NowRow,
NowCol, -1, 0) '上
    lngReverse = lngReverse + lngRetCode
    lngRetCode = procCellsCheck(BoardStatusTemp(), CheckColor, NowRow,
NowCol, -1, 1) '右上
    lngReverse = lngReverse + lngRetCode
    lngRetCode = procCellsCheck(BoardStatusTemp(), CheckColor, NowRow,
NowCol, 0, -1) '左
    lngReverse = lngReverse + lngRetCode
    lngRetCode = procCellsCheck(BoardStatusTemp(), CheckColor, NowRow,
NowCol, 0, 1) '右
    lngReverse = lngReverse + lngRetCode
    lngRetCode = procCellsCheck(BoardStatusTemp(), CheckColor, NowRow,
NowCol, 1, -1) '左下
```

```
    lngReverse = lngReverse + lngRetCode
    lngRetCode = procCellsCheck(BoardStatusTemp(), CheckColor, NowRow,
NowCol, 1, 0) '下
    lngReverse = lngReverse + lngRetCode
    lngRetCode = procCellsCheck(BoardStatusTemp(), CheckColor, NowRow,
NowCol, 1, 1) '右下
    lngReverse = lngReverse + lngRetCode

    procReverseCheck = lngReverse
End Function

'指定された方向について、返すコマの数を調べる
Private Function procCellsCheck(BoardStatusTemp() As Long, CheckColor As
Long, _
                                NowRow As Long, _
                                NowCol As Long, _
                                RowStep As Long, _
                                ColStep As Long) As Long
    Dim lngTempRow As Long  '一時行カウンタ
    Dim lngTempCol As Long  '一時列カウンタ
    Dim lngRevCtr  As Long  '返せるコマの数
    Dim lngReverseValue As Long
    Dim lngSameValue As Long

    Select Case CheckColor  '調べる色の種類
    Case VAL_BLACK  '黒の場合
        lngSameValue = VAL_BLACK    '同じ色は黒
        lngReverseValue = VAL_WHITE '返す色は白
    Case VAL_WHITE  '白の場合
        lngSameValue = VAL_WHITE    '同じ色は白
        lngReverseValue = VAL_BLACK '返す色は黒
    End Select

    lngRevCtr = 0
    lngTempRow = NowRow
    lngTempCol = NowCol

    Do
        lngTempRow = lngTempRow + RowStep
        lngTempCol = lngTempCol + ColStep
        Select Case BoardStatusTemp(lngTempRow, lngTempCol)
        Case lngReverseValue '置くコマと違う色の場合
            lngRevCtr = lngRevCtr + 1
        Case lngSameValue   '置くコマと同じ色の場合
            If lngRevCtr > 0 Then   'カウントアップ
                procCellsCheck = lngRevCtr
            End If
            Exit Do
        Case VAL_SPACE  '空白の場合
            Exit Do
        End Select
    Loop

End Function


'指定されたマスにコマを置き、コマを返す
Private Sub procReverse(BoardStatusTemp() As Long, RevColor As Long, NowRow
As Long, NowCol As Long)
    Dim lngTempRow As Long  '一時行カウンタ
    Dim lngTempCol As Long  '一時列カウンタ
    Dim lngTempCtr As Long  '一時カウンタ
    Dim lngRevCtr  As Long  '反転カウンタ
    Dim lngRowStep As Long  '行方向
    Dim lngColStep As Long  '列方向
    Dim lngSameValue As Long    '置いたコマと同じ色
    Dim lngReverseValue As Long '返すコマのいろ

    Select Case RevColor  '色の種類
    Case VAL_BLACK  '黒の場合
        lngSameValue = VAL_BLACK    '同じ色は黒
        lngReverseValue = VAL_WHITE '返す色は白
    Case VAL_WHITE  '白の場合
        lngSameValue = VAL_WHITE    '同じ色は白
        lngReverseValue = VAL_BLACK '返す色は黒
    End Select


    '指定されたマスにコマを置く
    BoardStatusTemp(NowRow, NowCol) = RevColor
    plngTurnCount = plngTurnCount + 1
    Call procDrawPiece

    If RevColor = plngPlayerColor Then
```

➡ つぎのページに続く

```
        MsgBox "プレイヤーの手は" & CStr(NowRow) & "行  " & CStr(NowCol) &
"列です。"
    Else
        MsgBox "コンピューターの手は" & CStr(NowRow) & "行  " &
CStr(NowCol) & "列です。"
    End If

    'コマを返す
    Call procReverseBoard(BoardStatusTemp(), RevColor, NowRow, NowCol)

End Sub

'配列上でコマを返す処理
Private Sub procReverseBoard(BoardStatusTemp() As Long, RevColor As Long,
RevRow As Long, RevCol As Long)
    Dim lngTempRow As Long  '一時行カウンタ
    Dim lngTempCol As Long  '一時列カウンタ
    Dim lngTempCtr As Long  '一時カウンタ
    Dim lngRevCtr  As Long  '反転カウンタ
    Dim lngRowStep As Long  '行方向
    Dim lngColStep As Long  '列方向
    Dim lngSameValue As Long    '置いたコマと同じ色
    Dim lngReverseValue As Long '返すコマのいろ


    Select Case RevColor  '色の種類
    Case VAL_BLACK  '黒の場合
        lngSameValue = VAL_BLACK    '同じ色は黒
        lngReverseValue = VAL_WHITE '返す色は白
    Case VAL_WHITE  '白の場合
        lngSameValue = VAL_WHITE    '同じ色は白
        lngReverseValue = VAL_BLACK '返す色は黒
    End Select

    BoardStatusTemp(RevRow, RevCol) = RevColor

    '8方向について返すコマを調べる
    For lngRowStep = -1 To 1
        For lngColStep = -1 To 1
            lngRevCtr = 0
            lngTempRow = RevRow
            lngTempCol = RevCol

            Do
                lngTempRow = lngTempRow + lngRowStep
                lngTempCol = lngTempCol + lngColStep
                Select Case BoardStatusTemp(lngTempRow, lngTempCol)
                Case lngReverseValue '違う色の場合
                    lngRevCtr = lngRevCtr + 1
                Case lngSameValue   '同じ色の場合
                    If lngRevCtr > 0 Then   '返すコマがあったら
                        'コマを返す
                        For lngTempCtr = 1 To lngRevCtr
                            BoardStatusTemp(RevRow + lngRowStep *
lngTempCtr, _
                                    RevCol + lngColStep * lngTempCtr) =
lngSameValue
                        Next lngTempCtr
                    End If
                Exit Do
                Case VAL_SPACE '空白の場合
                    Exit Do
                End Select
            Loop
        Next lngColStep
    Next lngRowStep

End Sub


'盤面をコピーする処理
Private Sub procCopyBoard(BoardStatusFrom() As Long, BoardStatusTo() As
Long)
    Dim lngRowCtr As Long
    Dim lngColCtr As Long

    For lngRowCtr = 0 To 9
        For lngColCtr = 0 To 9
            BoardStatusTo(lngRowCtr, lngColCtr) =
BoardStatusFrom(lngRowCtr, lngColCtr)
        Next lngColCtr
    Next lngRowCtr
End Sub'
```

《リスト3》 ミニ・マックス法でN手先まで読む(変更部分のみ)

```
'コンピュータの手を決める処理(変更)
Private Sub procCompIdea(IdeaCount As Long)
    Dim lngCompIdea As Long 'コンピューターが打てる手の場所数
    Dim lngWorkCtr As Long 'ワークカウンタ
    Dim lngMuchCtr As Long '高ポイント
    Dim lngMuchIdea(64) As Long '高ポイント格納用配列
    Dim lngMaxValue As Long '最高ポイント
    Dim lngPointValue As Long  'マスのポイント
    Dim lngNextCount As Long
    Dim lngMaxPoint As Long

    Application.Cursor = xlWait 'カーソルを砂時計にする
    lngMaxValue = 0
    lngMuchCtr = 1
    lngMuchIdea(1) = 0

    If plngTurnCount >= 55 Then
        lngNextCount = 61 - plngTurnCount
    Else
        lngNextCount = 5
    End If

    'lngNextCount手先まで読む
    lngCompIdea = procMinMax(BoardStatus(), _
                plngCompColor, _
                IdeaCount, _
                lngMaxPoint, _
                udtIdea(), _
                lngNextCount)

    'コマを返す処理
    Call procReverse(BoardStatus(), _
                plngCompColor, _
                udtIdea(lngCompIdea).Row, _
                udtIdea(lngCompIdea).Col)
    Call procDrawPiece '盤面を描く

    Application.Cursor = xlDefault 'カーソルを砂時計にする

End Sub

'MinMax法でNextCount手先まで読む(変更)
Private Function procMinMax(BoardStatusTemp() As Long, _
                CheckColor As Long, _
                IdeaCount As Long, _
                MaxPoint As Long, _
                udtIdeaNext() As typIdea, NextCount As Long) As
 Long

    Dim BoardStatusNext(9, 9) As Long  '手を打った後の盤面
    Dim udtIdeaNNext(64) As typIdea    '次ぎの手を格納するユーザ定義型
    Dim lngWorkCtr As Long          'ワークカウンタ
    Dim lngNNextCtr As Long         'ワークカウンタ(次の手)
    Dim lngMuchCtr As Long          '最高ポイント用カウンタ
    Dim lngMuchIdea(64) As Long       '高ポイント格納用配列
    Dim lngMaxValue As Long          '最高ポイント
    Dim lngNMaxValue As Long          '最高ポイント
    Dim lngMinValue As Long          '最低ポイント
    Dim lngPointValue As Long         'マスのポイント
    Dim lngNNextIdea As Long          '次ぎの手の数
    Dim lngNextCount As Long          '再帰回数
    Dim lngMinIdea As Long          '次ぎの次の手の最善手
    Dim lngNextCheckColor As Long       '調べる色

    lngMaxValue = -64
    lngMuchCtr = 1
    lngMuchIdea(1) = 0

    If NextCount = 0 Then
        procMinMax = 0
        Exit Function
    End If
```

つぎのページに続く

```
  lngNextCount = NextCount - 1
  '次の手すべてについて調べる
  For lngWorkCtr = 1 To IdeaCount
    '盤面を複写する
    Call procCopyBoard(BoardStatusTemp(), BoardStatusNext())
    '次の手を盤面に反映する

    Call procReverseBoard(BoardStatusNext(), _
                CheckColor, _
                udtIdeaNext(lngWorkCtr).Row, _
                udtIdeaNext(lngWorkCtr).Col)

    lngNNextIdea = -1 '変数の初期化
    '次ぎの次の手がいくつあるか調べる
    If CheckColor = plngCompColor Then
      lngNextCheckColor = plngPlayerColor
    Else
      lngNextCheckColor = plngCompColor
    End If
    lngNNextIdea = procIdeaCheck(BoardStatusNext(), lngNextCheckColor,
udtIdeaNNext())

    If lngNNextIdea = 0 Then '次ぎの次の手が無いなら
      udtIdeaNext(lngWorkCtr).Pnt = 8  '良い手とする
      lngNextCount = 0

    Else
      lngNMaxValue = 0 '変数の初期化
      '次ぎの次の手の最善手を調べる
```

```
      lngMinIdea = procMinMax(BoardStatusNext(), _
                   lngNextCheckColor, _
                   lngNNextIdea, _
                   lngMinValue, _
                   udtIdeaNNext(), _
                   lngNextCount)

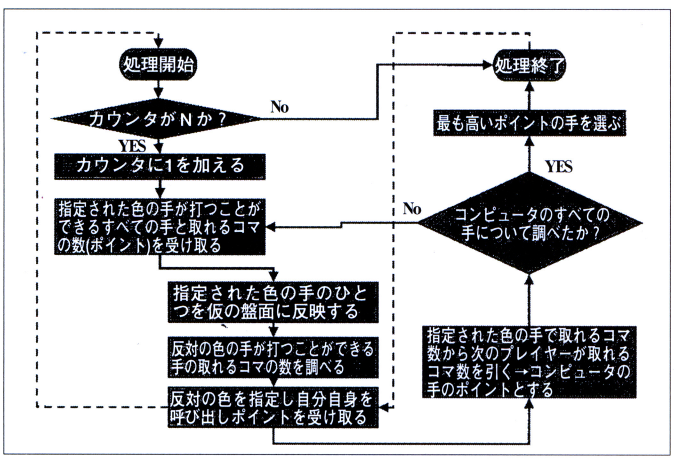
      '次の次の手の最善手を次の手の結果から引く
      udtIdeaNext(lngWorkCtr).Pnt = udtIdeaNext(lngWorkCtr).Rev
- lngMinValue

    End If

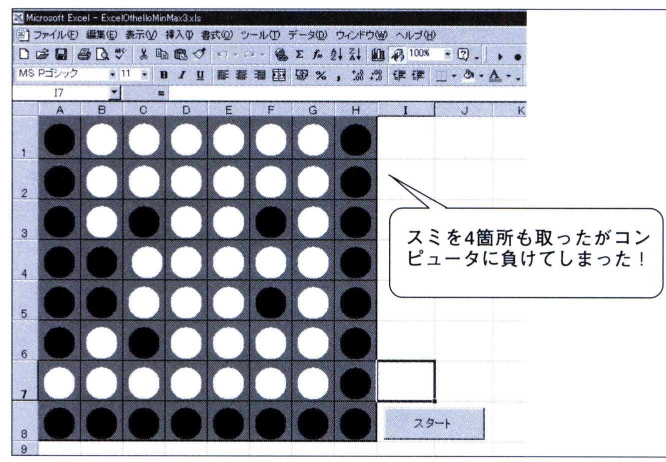
    Select Case udtIdeaNext(lngWorkCtr).Pnt  '次の手(Max)のポイント
    Case Is = lngMaxValue '最大ポイントと同じ
      lngMuchCtr = lngMuchCtr + 1
      lngMuchIdea(lngMuchCtr) = lngWorkCtr '手の番号を格納
    Case Is > lngMaxValue '最大ポイントより大きい場合
      lngMuchCtr = 1
      lngMaxValue = udtIdeaNext(lngWorkCtr).Pnt '最大ポイントを保存
      lngMuchIdea(1) = lngWorkCtr '手の番号を格納
    Case Else
    End Select
  Next lngWorkCtr

  MaxPoint = lngMaxValue
  '最高ポイントの手が複数ある場合ランダムに手を選択する
  procMinMax = lngMuchIdea(Int((lngMuchCtr * Rnd) + 1))

End Function
```



《第12図》再帰プロシージャーをＮ手先まで読む流れ図



《第13図》評価テーブルとミニ･マックス法をミックスした実行結果

ストはリスト2です。前回と重複する部分もありますが，すべて紹介します(長くてすみません)。実行してみると……。2手先までしか読んでいませんので，あまり強くはないですね。このプログラムを元に改善する必要があります。

再帰プロシージャーを使って改造する

先を読むコツがわかったところで，先読み部分を再帰プロシージャーに変更し，指定した先まで読むようにしてみましょう(リスト3)。先読みの回数を判断することと，色が交互に変わることに気を付けてください。この部分はプログラムを動かしながらじっくり考える必要があります。流れ図(第12図)を見ながら分析してみてください。

評価テーブルとミニ・マックス法をミックス

さて，ミニ・マックス法でのプログラムが動作するようになりましたが，処理時間を考えると4～5手先までしか先読みできないと思います。ちなみに，筆者の環境(CPU:Pentium4 733MHz, メモリ：312MB)では，5手先まで読んで大体10～30秒ほどかかります。

また，特に中盤で，セオリーから外れた変な手を打つなど，決して強いわけではありません。まだまだ人間有利です。したがって，段階ごとにセオリーも取り入れる必要がありあます。

手が進むにつれてのセオリーとして，次のようなものがあります。

・序盤～中盤はコマの数よりコマの位置が重要

・中盤～終盤はスミが重要

・終盤のミスは致命的

このセオリーを活かすために，前回試した盤面の評価テーブルを利用することにします。

・序盤は評価テーブルのみ

・中盤～終盤は評価テーブルとミニ・マックス法のミックス

・最後の詰めはミニ・マックス法のみ

といった具合に，段階的に作戦を変更することにします。今回はリストだけ紹介しま

すので（**リスト4**），ぜひ試してみてください(**第13図**)。特に終盤で，コンピューターが強さを発揮するはずです。また，誌面の都合でα-β法は紹介できませんが，改造にもチャレンジしてください。

まとめ

パズル・ゲームをテーマに6回にわたり連載しましたが，いかがだったでしょうか？6回という短い間では紹介しきれない部分もありますが，またの機会に紹介することにしましょう。 次回からは，新シリーズです。初心者での楽しめるように基本から紹介します。お楽しみに。

《リスト4》 評価テーブルとミニ・マックスをミックスする(変更・追加部分)

```
'コンピュータの手を決める処理(変更)
Private Sub procCompIdea(IdeaCount As Long)
  Dim lngCompIdea As Long 'コンピューターが打てる手の場所数
  Dim lngWorkCtr As Long 'ワークカウンタ
  Dim lngMuchCtr As Long '高ポイント
  Dim lngMuchIdea(64) As Long '高ポイント格納用配列
  Dim lngMaxValue As Long '最高ポイント
  Dim lngPointValue As Long  'マスのポイント
  Dim lngNextCount As Long
  Dim lngMaxPoint As Long
  Dim lngMinMaxPoint As Long


  Application.Cursor = xlWait 'カーソルを砂時計にする
  lngMaxValue = -100
  lngMuchCtr = 1
  lngMuchIdea(1) = 0

  For lngWorkCtr = 1 To IdeaCount
    Select Case plngTurnCount  'ターン数
    Case 1 To 22  '序盤
      lngPointValue = PointTable1(udtIdea(lngWorkCtr).Row, _
                    udtIdea(lngWorkCtr).Col)

    Case 23 To 40  '中盤
      lngNextCount = 3
      lngPointValue = PointTable2(udtIdea(lngWorkCtr).Row, _
                    udtIdea(lngWorkCtr).Col)
      Call procEvalMinMax(BoardStatus(), _
                plngCompColor, _
              udtIdea(lngWorkCtr).Row, _
              udtIdea(lngWorkCtr).Col, _
              lngMinMaxPoint, _
              lngNextCount)
      lngPointValue = PointTable1(udtIdea(lngWorkCtr).Row, _
                    udtIdea(lngWorkCtr).Col)
      lngPointValue = 4 * lngPointValue + lngMinMaxPoint

    Case 41 To 51  '終盤
     lngNextCount = 4
     lngPointValue = PointTable3(udtIdea(lngWorkCtr).Row, _
                   udtIdea(lngWorkCtr).Col)
      Call procEvalMinMax(BoardStatus(), _
                plngCompColor, _
              udtIdea(lngWorkCtr).Row, _
              udtIdea(lngWorkCtr).Col, _
              lngMinMaxPoint, _
              lngNextCount)
      lngPointValue = PointTable1(udtIdea(lngWorkCtr).Row, _
                    udtIdea(lngWorkCtr).Col)
      lngPointValue = 5 * lngPointValue + lngMinMaxPoint

    Case 52 To 60  '詰め
     lngNextCount = 61 - plngTurnCount
     lngPointValue = PointTable3(udtIdea(lngWorkCtr).Row, _
                   udtIdea(lngWorkCtr).Col)
      Call procEvalMinMax(BoardStatus(), _
                plngCompColor, _
              udtIdea(lngWorkCtr).Row, _
              udtIdea(lngWorkCtr).Col, _
              lngMinMaxPoint, _
              lngNextCount)
      lngPointValue = lngMinMaxPoint
    Case Else
    End Select

    Select Case lngPointValue  'マスのポイント
    Case Is = lngMaxValue '最大ポイントと同じ
      lngMuchCtr = lngMuchCtr + 1
      lngMuchIdea(lngMuchCtr) = lngWorkCtr '手の番号を格納
    Case Is > lngMaxValue '最大ポイントより大きい場合
      lngMuchCtr = 1
      lngMaxValue = lngPointValue '最大ポイントを保存
      lngMuchIdea(1) = lngWorkCtr '手の番号を格納
    Case Else
    End Select
  Next lngWorkCtr

  '最高ポイントの手が複数ある場合ランダムに手を選択する
  lngCompIdea = lngMuchIdea(Int((lngMuchCtr * Rnd) + 1))

  'コマを返す処理
  Call procReverse(BoardStatus(), plngCompColor,
udtIdea(lngCompIdea).Row, udtIdea(lngCompIdea).Col)
  Call procDrawPiece '盤面を描く
  Application.Cursor = xlDefault 'カーソルを戻す

End Sub

'指定された手から派生する手をMinMax法で調べる(追加)
Private Sub procEvalMinMax(BoardStatusTemp() As Long, _
            CheckColor As Long, _
            NowRow As Long, _
            NowCol As Long, _
            MaxPoint As Long, _
            NextCount As Long)
  Dim BoardStatusNext(9, 9) As Long  '手を打った後の盤面
  Dim udtIdeaNNext(64) As typIdea   '次ぎの手を格納するユーザ定義型
  Dim lngWorkCtr As Long        'ワークカウンタ
  Dim lngNNextCtr As Long       'ワークカウンタ(次の手)
  Dim lngMuchCtr As Long        '最高ポイント用カウンタ
  Dim lngMuchIdea(64) As Long     '高ポイント格納用配列
  Dim lngMaxValue As Long       '最高ポイント
  Dim lngNMaxValue As Long       '最高ポイント
  Dim lngMinValue As Long       '最低ポイント
  Dim lngPointValue As Long      'マスのポイント
  Dim lngNNextIdea As Long       '次ぎの手の数
  Dim lngNextCount As Long       '再帰回数
  Dim lngMinIdea As Long        '次ぎの次の手の最善手
  Dim lngNextCheckColor As Long     '調べる色

  lngMaxValue = -100
  lngMuchCtr = 1
  lngMuchIdea(1) = 0

  lngNextCount = NextCount - 1
  '次の手すべてについて調べる

  '盤面を複写する
  Call procCopyBoard(BoardStatusTemp(), BoardStatusNext())
  '次の手を盤面に反映する

  Call procReverseBoard(BoardStatusNext(), _
            CheckColor, _
            NowRow, _
            NowCol)

  lngNNextIdea = -1 '変数の初期化
  '次ぎの次の手がいくつあるか調べる
  If CheckColor = plngCompColor Then
    lngNextCheckColor = plngPlayerColor
  Else
    lngNextCheckColor = plngCompColor
  End If
  lngNNextIdea = procIdeaCheck(BoardStatusNext(), lngNextCheckColor,
udtIdeaNNext())

  If lngNNextIdea = 0 Then '次ぎの次の手が無いなら
    MaxPoint = 8  '良い手とする
    lngNextCount = 0
  Else
    lngNMaxValue = 0 '変数の初期化
    '次ぎの次の手の最善手を調べる
    lngMinIdea = procMinMax(BoardStatusNext(), _
              lngNextCheckColor, _
              lngNNextIdea, _
              lngMinValue, _
              udtIdeaNNext(), _
              lngNextCount)

     '次の次の手の最善手を次の手の結果から引く
     MaxPoint = 0 - lngMinValue
  End If
End Sub
```

21世紀の遊びはコレ！ロボ道場

AIBOをプログラミングしよう!!

「AIBOマスタースタジオ version1.1」

取材／文　松原拓也



◀「AIBOマスタースタジオ version 1.1」（ERS-210/220/311/312 用）
対応機種：Windows 98 SE/Me/2000 Professional/XP
価格：50,000円（税別）／発売中
WEBサイト　http://www.jp.aibo.com/ams/
＊メモリースティックの読み書き可能なパソコンが必要です。

マスタースタジオがバージョンアップ

ソニーのエンタテインメントロボット「AIBO」のプログラミング・ツール「AIBOマスタースタジオ」が「version1.1」になりました。今回のバージョンアップでは，以下の変更がされています。

①ERS-210以外にも対応

「ERS-220」や「ERS-311/312（LATTE＆MACARON）」にも対応しました。

②音声認識への対応

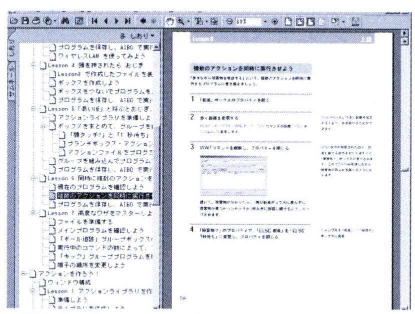
対応している音声の数は約50種類。その他，「ピポパ」音のAIBOサウンドにも対応しました。

③マニュアルの充実

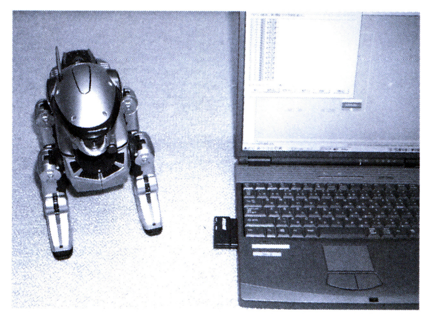
「チュートリアルガイド」が大幅にボリュームアップして分かりやすくなりました（**写真1**）。サンプルプログラムは初心者向けから，同時にアクションを行うような高度なものまで広範囲にサポートしています。

④操作性の向上

ワイヤレスLAN接続時にプログラム内のデータが表示されるなど，利便性が増しています（**写真2**）。

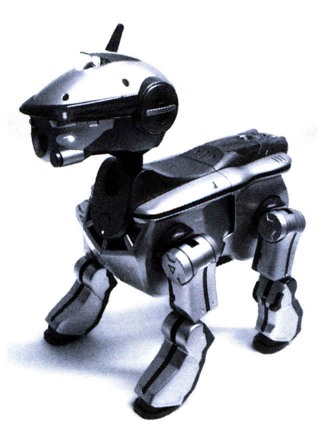


《写真1》いっそう充実したマニュアル。「ユーザーガイド」と「チュートリアルガイド」



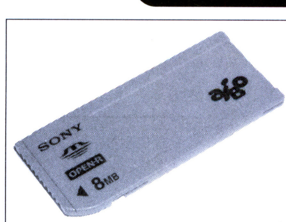
《写真2》無線LANで効率的なプログラミングが可能

エンタテイメントロボットAIBO



▲近未来的なデザインが採用された「AIBO ERS-220」
価格：180,000円（税別）／発売中
ソニー
WEBサイト　http://www.aibo.com

ＡＩＢＯのアクセサリ



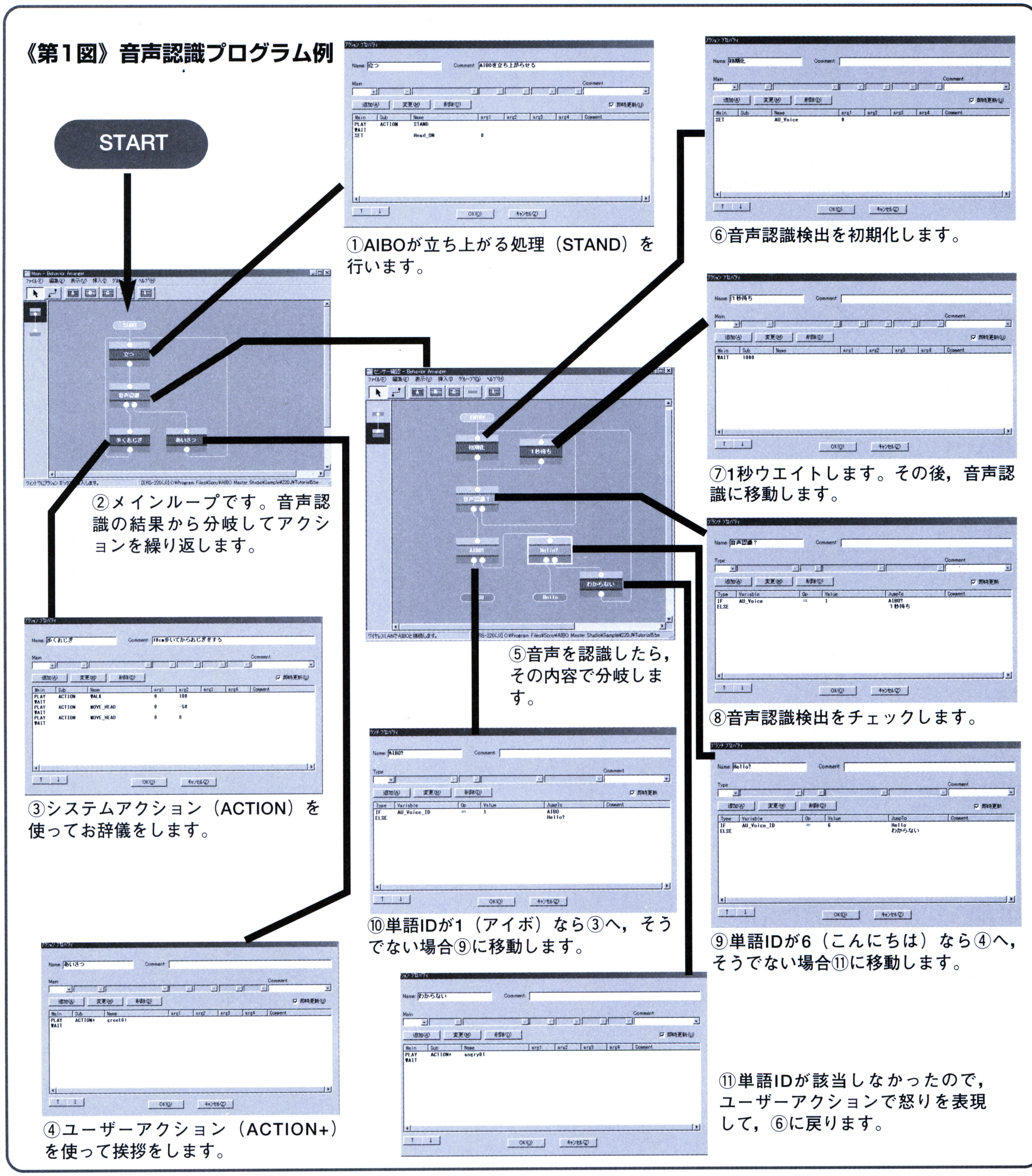
▲「AIBOプログラミングメモリースティックERA-MS008」。3,400円（税別）。AIBOに装着する専用のメモリースティック（「AIBOマスタースタジオ」に1枚付属）



▲「AIBOワイヤレスLANカードERA-201D1」。20,000円（税別）。ERS-210/220用の無線LANカード



▲「USB対応メモリースティックリーダー／ライターMSAC-US1」。6,800円（税別）



《第1図》音声認識プログラム例

音声認識プログラム

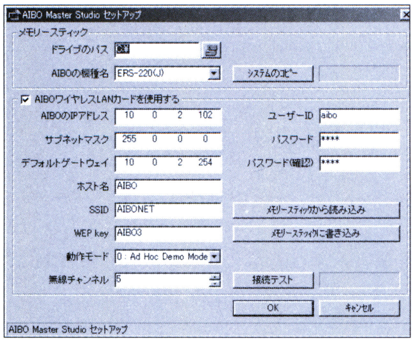
第1図は，AIBOが音声認識をするプログラムの例です（チュートリアルガイド「ビヘイビアを作ろう！編 Lesson5」より）。プログラムはアクション（動作）とビヘイビア(行動）で構成されています。ビヘイビアは「Behavior Window(ビヘイビア ウィンドウ)」と呼ばれる基本画面で設定できます。「何が起きたらどうする」といったフローチャート感覚でプログラミングできます。(**第1図**参照）。

プログラムの実行

第1図のプログラムをAIBOに実行させてみましょう。

「AIBOマスタースタジオ」は，「AIBOワイヤレスLANカード ERA-201D1」(別売）を使うことで，離れた場所からの効率的なプログラミングが可能です。

無線LANを使う場合，まずセットアップ画面でAIBOに固定のIPアドレスを割り振ります（**写真3**)。なお，パソコン側にはIEEE802.11b準拠の無線LAN PCカードか，無線LANのアクセスポイントが



《写真3》「AIBO Master Studio セットアップ」画面

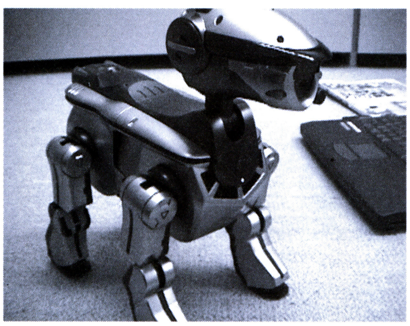


《写真4》ワイヤレスLAN（無線LAN）によるプログラムの実行画面

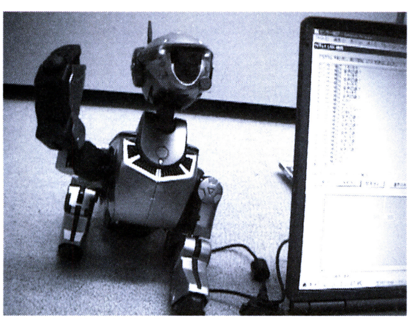
必要です。

「ワイヤレスLAN接続」というメニューを選択して，「実行」ボタンを押すと，作成されたプログラムが「R-CODE」というスクリプトに変換されて転送されます（**写真4**）。メモリースティックへの保存も可能です。

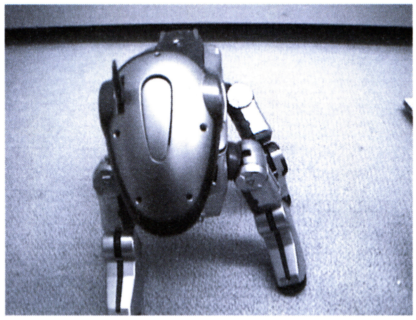
AIBOに向かって，「アイボ」や「こんにちは」と言うとアクションを起こします（**写真5〜7**）。成功です。



《写真5》電源ONの姿勢から立ち上がったAIBO



《写真6》「こんにちは」の声に反応して，挨拶をするAIBO



《写真7》「アイボ」の声に反応して，お辞儀をするAIBO

アクションの作成

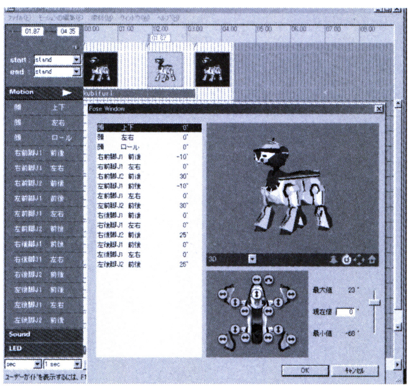
今度は，AIBOのビヘイビア（行動）を自作してみましょう。まずはアクションの作成です（**写真8〜10**）。

「Action Composer(アクション コンポーザー)」はAIBOのアクション（動作）を編集するツールです。AIBOの「LEDの表示」「サウンドの再生」「ポーズ」を，ひとつのアクションとして統合管理することができます。

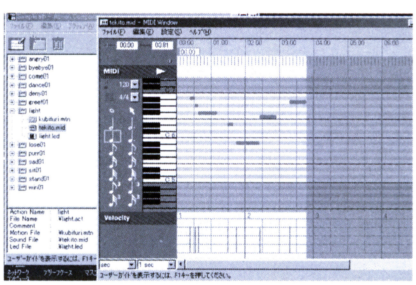
「ポーズ」とは「アクション ウィンドウ」のなかでAIBOが動くための中継点となる形のことです。各時点のポーズを設定すると，自動的に一連の動きとして補完されます。

サウンドはWAVとMIDI音の2種類を再生することができます。音声や効果音を再生するならばWAVファイル，音楽ならばMIDIファイルが適しています。

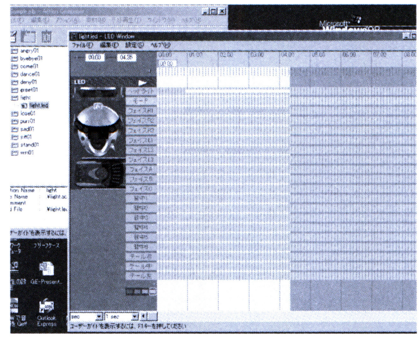
最後にライブラリを転送してアクションは完成です（**写真11**）。



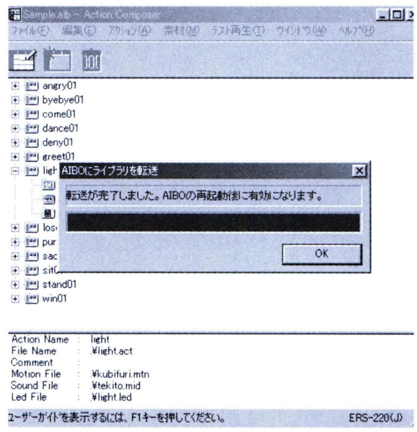
《写真8》「Action Window」でポーズを設定



《写真9》「MIDI Window」と「WAVE Window」でAIBOの発するサウンドを設定



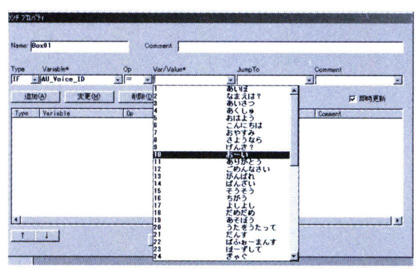
《写真10》「LED Window」で点灯するLEDのタイミングを設定



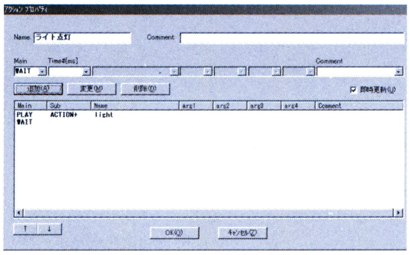
《写真11》新たに作成したライブラリをAIBOに転送する

ビヘイビアの作成

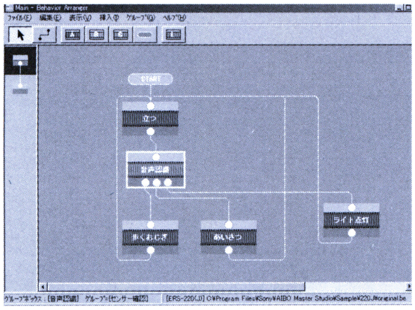
「Behavior Arranger(ビヘイビア アレンジャー)」でAIBOのビヘイビアを作成します。ボックスには，アクションを行なう「動作ボックス」，条件判定を行なう「分岐ボックス」，プログラムをグル



《写真12》「AIBOマスタースタジオ」で認識できる音声の数は約50種類



《写真13》「おーい」という音声に反応する分岐ブロックを追加



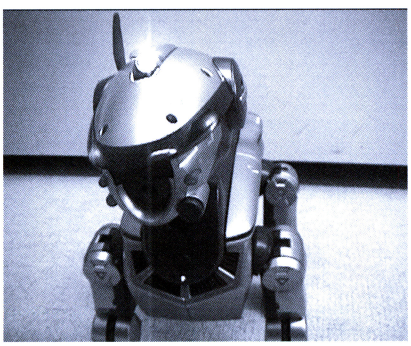
《写真14》音声に反応するアクションを追加

ープ化した「グループボックス」の3種類があります。アクションには，指定方向に歩行させる“WALK”命令や，画像認識で物体を追う“TRACK_HEAD”命令などがあります。「アクション コンポーザー」で作ったオリジナルのアクションを実行させることができます。プログラミングは文法を覚える必要がなく，コンボボックスを選択していくだけで簡単です（**写真12〜14**）。

実 行

前回同様，無線LANでAIBOにプログラムを転送，実行させます。

AIBOに「おーい」と話しかけてみます。頭上のリトラクタブルライトが光りました。さらに音楽も再生され，首が動きます（**写真15**）。成功です!!



《写真15》「おーい」という声に反応するAIBO。ライトを点灯させ，首を動かしている



《写真16》「AIBOマスタースタジオ」のWEBサイト。R-CODEのシステムファイルやインタープリタを無償でダウンロードすることができます
http://www.jp.aibo.com/ams/

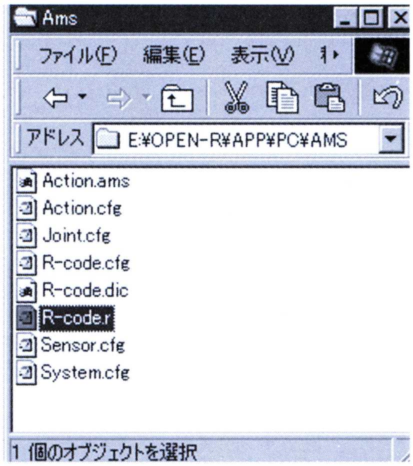
R-CODEに挑戦

次はR-CODEでAIBOをプログラミングしてみましょう。

「R-CODE」とは，AIBOをプログラミングできるインタープリタ用の簡易スクリプト言語です。センサー情報や変数，組み込みコマンドが中間言語の形でコード化されています。「R-CODE」用のシステムファイルとインタープリタは無償で配布されています（**写真16**）。

R-CODEを楽しむためには，「AIBO本体」と「AIBOプログラミング・メモリースティック」，そしてメモリースティックが読み書き可能なパソコンが必要です。

R-CODEはスクリプト言語なので，エディタや「メモ帳」でプログラムできます。R-CODEの書式はBASICのように親しみやすいです。コマンドリファレンスはR-CODEのサポートページ（http://www.jp.aibo.com/ams/rcode.html）からダウンロードできます。



《写真17》メモリースティック内の“OPEN-R¥APP¥PC¥AMS”の階層に“R-CODE.R”を保存する

R-CODEの実行

R-CODEをメモリースティックに書き込みます。メモリースティックにシステムファイルとインタープリタをコピーしておきましょう（**写真17**）。プログラムはメモリースティック内の“OPEN-R¥APP¥PC¥AMS”ディレクトリに“R-CODE.R”というファイル名で保存します（**リスト1**）。書き込みが終わったら，AIBOのメモリースティック・スロットにメモリースティックを装着します。AIBOの電源を入れると，プログラムが自動的に実行されます。

《リスト1》“OPEN-R¥APP¥PC¥AMS¥R-CODE.R”

```
//「前進」という音声に反応して前に進むプログラム
// R-CODE.R

  PLAY:ACTION:STAND //立ち上がり処理
  SET:AU_Voice:0    //音声検知クリア

  DO
    IF:AU_Voice:=:1:THEN  //音声認識検知
      WAIT:1
      IF:AU_Voice_ID:=:35:THEN  //ID=「前進」
        PLAY:ACTION:WALK:0:100  //前進
        WAIT
      ENDIF
      SET:AU_Voice:0  //音声検知クリア
    ELSE              //音声認識なしの場合
      WAIT:1000       //ウエイト
    ENDIF
  LOOP
```

最後に

自作のR-CODEとシステムファイル，インタープリタは自由に配布することができます（ソフトウェア使用許諾に準じた形で）。プログラムを公開しあうことで，アッと驚くプログラムが生まれるかもしれません。AIBOの楽しみ方が広がりそうです。

***今回作成したプログラムは本誌『マイコンBASICマガジン』のWEBサイトにてダウンロードすることができます。**

時代を先取り！ 目指せ最先端プログラマー！

.NET（ドットネット）プログラミング入門

第1回 .NETアプリケーション

廣瀬 貴一

Microsoft .NET

現在インターネットが多様化するなかで，Webサービスの統合に向けて，MicrosoftがXML Webサービスのために新しいプラットフォーム，**Microsoft .NET**（ドットネット）を提唱していることは，ネットや雑誌で見たことがあると思います。

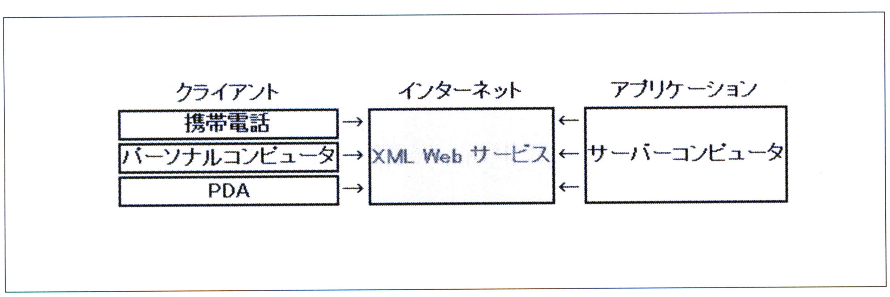
この連載では6回にわたって，開発者から見た技術的な内容で.NETを紹介したいと思います。これを理解してもらえれば，これまでは理屈のうえでの概要しかわからなかった.NETの本質や，技術的な実装方法を知ることができるでしょう。

.NETプラットフォームとXML Webサービスによって，今までは実現できなかった強力なサービスを実現し，ネットワークだけではなく，ネットワークを用いた実世界のビジネスにも大きな可能性を生み出すことができます。

これまで，アプリケーションはシステムのために存在し，システムのために記述されてきましたが，XML Webサービスの実現によって，アプリケーションはシステムやプログラミング言語のような環境の制限を超え，それらの種類を問わずにインターネットを経由して通信を行うことが可能となります。

例えば，.NET My Servicesというテクノロジーでは，クライアントコンピュータはオープンな認証システムをもっているため，ユーザーはこれに一度ログオンすれば，ネット上のWebやアプリケーションにアクセスするために，IDやパスワードを入力するようなことから解放されます（**第1図**）。

現在のWebサイトはそれぞれが孤島のように孤立し，ユーザーはそれぞれのサービスを受けるために，それぞれのWebサイトにアクセスし，必要な情報を送信しなければなりません。しかし，XML Webサービスは，ユーザーにより良いサービスを提供するために，お互いに協調・統合することができるようになります。



《第1図》XML Webサービスの概念図

.NET Framework

開発者にとって重要なのは.NETアプリケーションは論理的に，OS依存せずに動作することができるという点です。つまり，OSに関係なく動作するため，一度.NETアプリケーション用のソースを記述すれば，移植作業をすることなく他のOSでも動かすことができるのです。

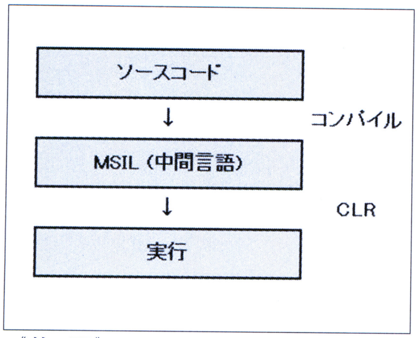
これによって，ネットワーク上のコンピュータが連携したサービスを可能とし，より高度なソリューションが実現できるということが.NETの主張なのです。

では，どうやって.NETプラットフォームはOSを抽象化しているのでしょう？

.NET開発環境はソースをコンパイルすると機械語ではなく.NET専用の中間言語**MSIL（Microsoft Intermediate Language）**を生成します。MSILは機械語ではないので，この状態ではまだ実行することができません

これを，.NETの中枢技術といえる**CLR（Common Language Runtime）**が読みこみ，JITコンパイラと呼ばれるものでMSILがネイティブな機械語に変換されて実行されます。現在JITコンパイラはインストール時にコンパイルするもの，実行時にコンパイルするものなど数種類あります。

通常はアプリケーション実行時（正確



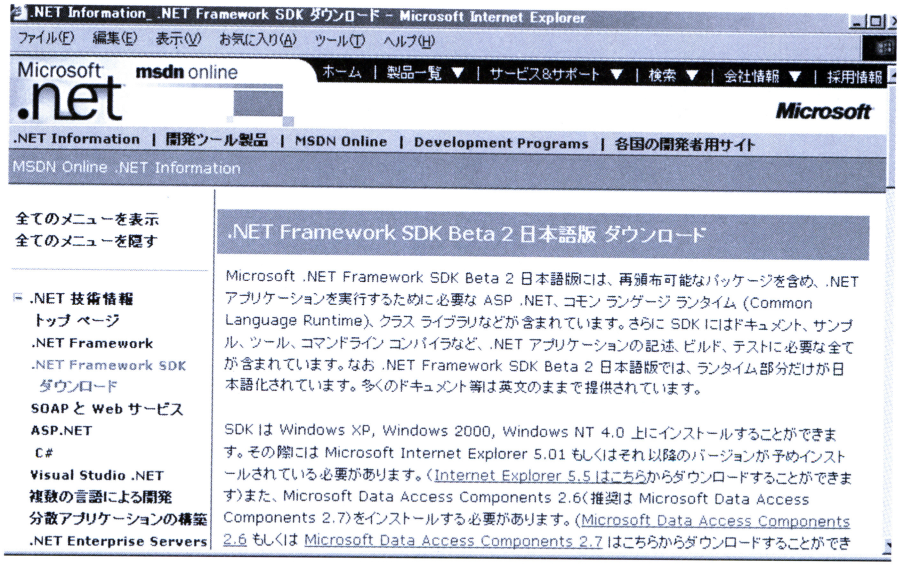
《第2図》MSILはCLR専用のマシン語!?

にはメソッド実行時）にコンパイルしてメモリに配置しますが，インストールアプリケーションならばインストール時にコンパイルしたほうがオーバーヘッドを減少させることができるでしょう（**第2図**）。

MSILはCLR専用のマシン語のようなもので，図を見ればCLRが**仮想マシン**の役割を果たしてOSを抽象化していることがわかります。MSILの仕様を理解していれば，コンパイルせずに直接MSILを記述して.NETアプリケーションを作ることも可能ですが，あなたが独自の.NET開発環境を作るような場合を除いて，まずそのようなことをする必要はありません。

.NET 開発環境

.NETアプリケーションの開発には，**.NET Framework SDK**と呼ばれる開発環境を用います。これは現在も開発中の段階ですがベータ版をMicrosoftの**http://www.microsoft.com/japan/developer/net/dotnetsdkdown.asp**からダウンロードすることが可能です（**写真1**）。



《写真1》.NET Framework SDKのベータ版がダウンロードできる

ダウンロードの方法は，一括でダウンロードする方法と，分割でダウンロードする方法があります。

一括の場合127MBもの容量があるので，ダウンロード支援ソフトウェアで行うことを奨めます。分割の場合は10.9MBの分割ファイルをすべてダウンロードして，ダウンロードしたファイルをすべて適当なフォルダに格納してください。作成したフォルダにすべての分割ファイルとetup.batが存在することを確認したら，setup.batを実行してください。setup.exeが自動的に作成されます。

後は，setup.exeを実行し，インストーラーにしたがってインストールしてください。インストール時の設定は，基本的にインストーラーの初期設定のものでかまわないでしょう。

.NETに関連する情報や，開発環境の仕様など細かい情報もMicrosoftのサイトで確認することができるので，定期的にチェックすると良いでしょう。また，雑誌の付属CDなどで提供されているVisual Studio .NETのベータ版でも開発することができます。

.NET Framework SDKはGUIでの開発をサポートしないので，コンソールの知識がない方はVS .NETを使用してください。ただし，この場では.NET Framework SDKを使用していることを想定して話を進めます。より詳しいインストール方法や，他に必要なアプリケーションなどはMicrosoftのサイトで確認することができます。

.NET と C#

.NETアプリケーションの開発は言語を選びません。MSILを生成するコンパイラであれば，どの言語で開発されてもかまわないのです。

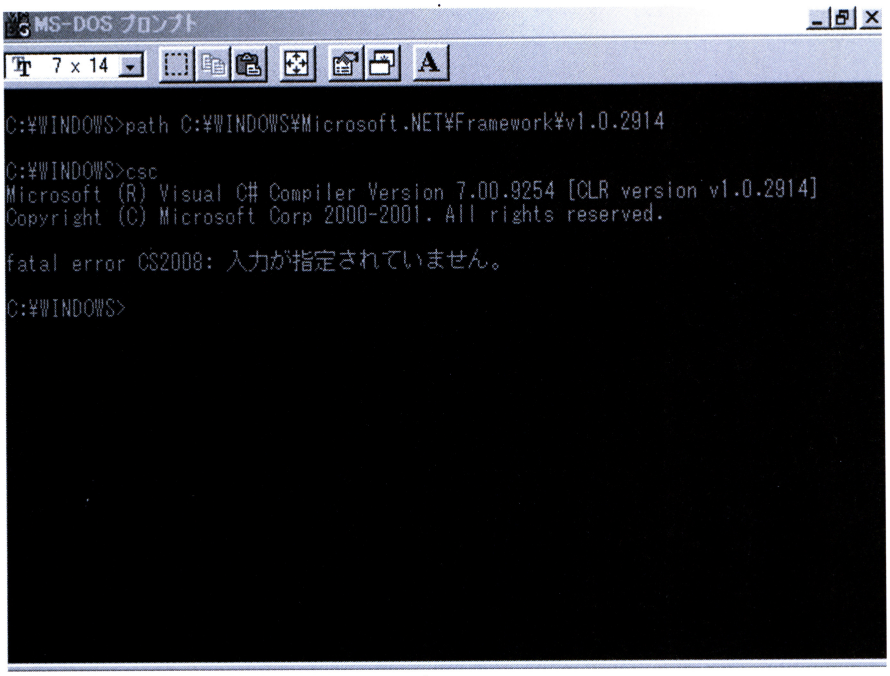
Microsoftは.NETアプリケーションを開発するコンパイラのために共通言語仕様CLS(Common Language Specification)を定義していて，コンパイラはこの規則に従う必要があり，これに従えばどのような開発環境であっても，従来の環境に互換性のある形で.NET開発環境を提供できるのです。

現在はVB .NETやJScript，C++などの，従来の言語のコンパイラがSDKに含まれており（ただし従来の言語もCLSに従う必要があるため一部の言語仕様は変更されている），さらにJava言語を基盤とした開発環境でJ#などというものも存在します。

しかし，.NETアプリケーション開発に最も適していると思われる言語は，Microsoftが提唱する**C#言語**です。C#は.NETのために生まれ，その仕様は.NET開発に最も適したオブジェクト指向言語となっています。

このコーナーでも，C#を中心に.NETアプリケーションを作ることにします。C#は，ベータ2日本語版のデフォルトインストールでC:¥WINDOWS¥Microsoft.NET¥Framework¥v1.0.2914フォルダにあるcsc.exeでコンパイルすることができます。

Visual Studioは総合開発環境ですが，.NET Framework SDKはコマンドラインツールのみです。MS-DOSプロンプトからコンパイルする時，どのフォルダからもコンパイラを見えるようにする必要があります。これを「パスを通す」と言いますが，コンパイラにパスを通すにはプロンプトで次の文を入力してください。



《写真2》.NET Framework SDKをコンパイル

path C:¥WINDOWS¥Microsoft.NET¥Framework¥v1.0.2914

path 以降はコンパイラが格納されているディレクトリを指定します。Windows を終了するとこの設定は消えてしまうので，起動後にまた入力する必要があります。

AUTOEXEC.BAT ファイルにパスを設定することで，Windows を起動時に自動的にパスを通す便利な方法もありますが，下手に編集すると Windows が起動しなくなるので注意が必要です。適当なフォルダで csc と入力して，Microsoft の著作権に関する文字列が出てくれば設定は成功です（**写真2**）。

パスが通れば，後は実際に C# のソースファイルを記述して csc コマンドでコンパイルするだけです。csc.exe は，基本的には次のように，コマンドとファイル名を指定することでコンパイルできます。

csc file-name

file-name にコンパイルするファイル名を指定してください。C# 言語のソースファイルの拡張子は ***.cs** です。オプションなどの細かい仕様は，SDK 付属のドキュメントなどを参照してください。

文字を出力

では，さっそくコンソールに文字を出力する基本的な .NET アプリケーションを C# 言語で生成してみましょう。

.NET アプリケーションを建築する場合，システムに関連する処理は .NET クラスライブラリを使用します。クラスライブラリは，すべてを網羅すれば本が1冊できるほど多種多様な機能をサポートしています。

この場で，クラスの仕様を詳細に解説することはできないので，クラスやそのメンバについては SDK 付属のドキュメントを参照してください。コンソールアプリケーションの標準入出力を行うには **System** 名前空間の **Console** クラスを用います。

public sealed class Console

名前空間の考え方は，基本的に Java のパッケージや C++ の名前空間と同じものと考えてかまいません。Console クラスの **Write()** メソッドを用いれば，コンソールに文字列を出力することができます。

出力文字の末尾に改行文字を含めるには**WriteLine()** メソッドを使ってください（**リスト1**）。

このプログラムをコンパイルすると EXE 実行ファイルが生成されるので実行してみてください。C# 言語は，まず静的な Main() メソッドをアプリケーションのエントリーポイントとして開始します。

大文字と小文字は区別されるので main() ではなく，最初の1文字目が大文字であることに注意してください（**写真3**）。

ファイルへの書き込み

Console クラスは，静的なメンバとして標準出力･入力エラーのストリームを保有しています。先ほどの Write() メソッドは，実はこのストリームに文字列を出力していたのです。標準出力は静的な Out 公開メンバを参照します。

Out メンバは C# 特有の **プロパティ** と呼ばれる，メンバ変数とアクセッサメソッドをワンセットにした特殊なメンバです。

public static TextWriter Out {get;}

Out は文字出力のための抽象クラス TextWriter 型のメンバ変数です。TextWriter クラスは System.IO 名前空間で宣言されています。

public abstract class TextWriter : MarshalByRefObject, IDisposable

このクラスは，文字をファイルに出力するための Write() メソッドと

《リスト1》EXE実行ファイル生成プログラム

```
class Test {
        static void Main() {
                System.Console.Write("Kitten on your lap");
        }
}
```



《写真3》リスト1を実行した画面

WriteLine() メソッドなどを提供していて，Console クラスの静的なWrite() メソッドやWriteLine() メソッドは，実際にはそれぞれのOut メンバ変数のメソッドを呼び出していたのです（**リスト2**）。

リスト1のプログラムと実行結果は同じですが，Console.Write() を用いずに標準出力のストリームであるTextWriter クラスの Write()メソッドを使用しています。C# では，クラスのインスタンスは常にヒープに生成され，クラス型変数は常にヒープの参照型となります。

そのため，Out を twWrite へ代入していますが，実際はインスタンスの参照を代入しただけで，インスタンスそのものが複製されるわけではありません。ヒープに生成されたインスタンスは，CLR が参照を監視し，インスタンスへの参照がなくなるとガーベジコレクタが自動的に開放してくれます。

このような方法でインスタンスを管理するという部分では，Java 仮想マシンと基本的に同じです。ストリームの概念がわかれば，論理的にコンソールの入出力もファイルの入出力も同じであるということが理解できます。

リスト2のプログラムの twWriteをコンソールの標準出力からファイルの出力ストリームに変更するだけで，簡単にファイルに文字列を書き込むことができるようになります。ディスクファイルの操作にはSystem.IO 名前空間の **File** クラスを使用します。

public sealed class File

このクラスはファイルのコピーや生成，削除などの操作をサポートしています。このクラスの静的メンバ CreateText (string path) メソッドを使えば，テキストファイルを生成することができます。CreateText() メソッドの第一引数には，ファイルのパスを文字列で指定します。

ファイルを生成すると同時に，メソッドは生成したファイルの **StreamWriter** を返します。StreamWriter クラスは抽象クラス TextWriter を継承したクラスで Write() メソッドなどをオーバーライドしています。

public class StreamWriter : TextWriter

TextWriter クラスを継承している StreamWriterクラスの変数は TextWriter 型の変数に暗黙のアップキャストを行うことができます。こうしてインスタンスを抽象化すれば，コンソールと同じ扱いでファイルに文字列を書き込むことができます。

たとえ型がキャストされても，メソッドはオーバーライドされているので実行時に適切なメソッドが実行されます。ただしファイルの場合は，ファイル操作を終了した時に **Close()** メソッドを呼び出してストリームを閉じる必要があります（**リスト3**）。
Close() メソッドはストリームを閉じる前にバッファをフラッシュします。

ただし，閉じる時以外に明示的にバッファをフラッシュしたい場合は **Flush()** メソッドを使ってください。

このプログラム1行目は，何度も System.IO 名前空間を指定するのが面倒なので using キーワードで名前空間を可視化しています。using は C++ では using namespace，Java ではimport にあたる文と考えてください。この文によって，System.IO 名前空間のシンボルを指定する時，名前空間を省略することができます。

このプログラムを実行すると，実行フ

《リスト2》Text WriterクラスのWrite ( ) メソッドを使用したプログラム

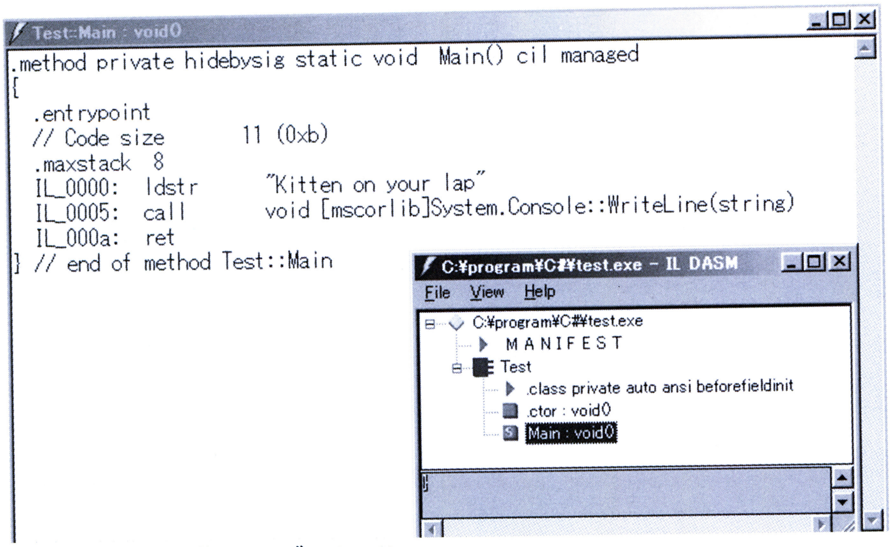
```
class Test {
        static void Main() {
                System.IO.TextWriter twWrite = System.Console.Out;
                twWrite.WriteLine("Kitten on your lap");
        }
}
```

《リスト3》ファイルに文字列を書き込むプログラム

```
using System.IO;

class Test {
        static void Main() {
                TextWriter twWrite = File.CreateText("test.txt");

                twWrite.Write("Kitten on your lap");
                twWrite.Close();
        }
}
```



《写真4》MSILの逆アセンブルルーツ

ファイルのディレクトリに test.txt というファイルが生成され，そのテキストファイルに，Write() メソッドで指定した文字列が書き込まれています。

MSIL の世界

.NET アプリケーションはネイティブな機械語ではなく，CLR が理解できる中間言語 MSILで構成されていると説明しましたが，実際にコンパイルするとネイティブな実行ファイルが生成されていました。この実行ファイルは特別なものではなく，Win32で標準的な PE ヘッダの実行ファイルです。

実はこのとき，MSIL は他の必要なデータと共にメタデータとして実行ファイルに埋め込まれているのです。この EXE コードを実行すると，通常の Windows アプリケーション同様に実行ファイルと必要なDLL がメモリにロードされ，JIT コンパイラでMSIL をネイティブコードにコンパイルしてから実行されます。

.NET Framework SDK には，デフォルトインストールで C:¥Program Files¥Microsoft.NET¥FrameworkSDK¥Bin フォルダに ildasm.exe という MSILの逆アセンブルツールが付属されています。

このツールを用いれば，.NET アプリケーションのMSIL を直接確認することができるので，先ほど作ったC# のプログラムが実際に MSILに変換された形を見てみましょう。

ツールの使い方は簡単で，メニューバーの File→ Open と選択して .NET Framework SDK で作成した実行ファイルを選択します（**写真4**）。

サンプルプログラム

最後に，今回の C# 言語で作った .NET アプリケーション を応用したサンプルを紹介します。このプログラムは，アプリケーション起動時にコンソールから指定した引数のファイルを開いて，それを表示するという簡単なテキストビューワプログラムです。

File クラスの OpenText(string path) メソッドは path で指定したパスのファイルを開き StreamReader を返します。

StreamReader クラスは，ストリームからデータを読み込むためのクラスです。このプログラムでは StreamReader クラスのReadToEnd() メソッドを使っています。

ReadToEnd() メソッドは，ファイルポインタ（このプログラムの場合はファイルの先頭）から最後までデータを読み込み，その string 型を返します（**リスト4**）。

テキストファイルを一度で読み込んでしまいたい場合に，このメソッドは非常に便利です。

＊現在の.NET Freamwork SDKはベータ版なので，クラスライブラリや言語仕様等は一部変更される可能性があります。

《リスト4》今月の.NETアプリケーション応用プログラム（テキストビューワプログラム）

```
using System;
using System.IO;

class Test {
        static void Main(string[] strArgs) {
                if (strArgs.Length == 0){
                        Console.WriteLine("引数にファイル名を指定してください");
                        return;
                }

                try {
                        StreamReader srFile = File.OpenText(strArgs[0]);
                        Console.WriteLine(srFile.ReadToEnd());
                        srFile.Close();
                }
                catch (FileNotFoundException err) {
                        Console.WriteLine("ファイルが見つかりません");
                        return;
                }
        }
}
```

自作一代記

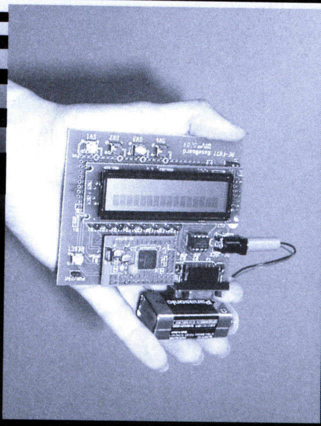
電子工作にチャレンジ！

連載第12回

マイコンでBASICインタプリタが動く

「AKI-PIC877モジュール用ベースボード・キット」

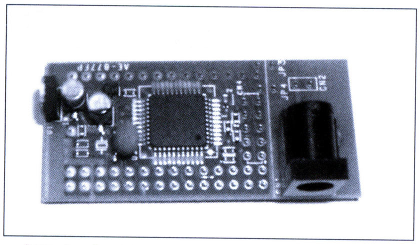
製作・文／松原拓也



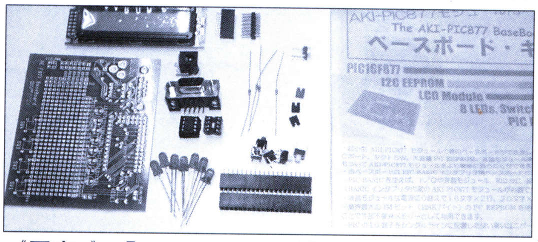
「AKI-PIC877モジュール用ベースボード・キット」は「AKI-PIC877モジュール」を使った汎用の電子工作キットです。なんとBASICインタプリターが動くそうです。

PIC搭載の超小型ボード

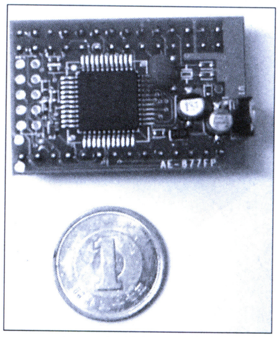
「AKI-PIC877モジュール用ベースボード・キット」（**写真1**）には「AKI-PIC877モジュール」という小型のマイコンボードが付属します（**写真2**）。プロセッサには8ビットマイコン「PIC」シリーズの「PIC16F877-20pt」が搭載されています（44ピン・フラットパッケージ）。動作クロックは20MHzで，PICでも最高クラスの性能です。そして，驚くのはこの基板の小ささです。しかもこの上には，セラロックと，三端子レギュレータ，RS232Cの通信用ICが搭載されています（部品は実装済みです）。なので，7～12Vの電圧さえ加えてあげれば，そのまま動作させることが可能です。



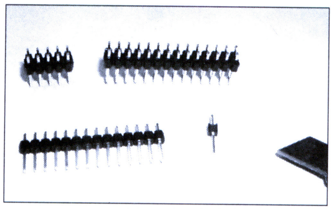
《写真2》付属する「AKI-PIC877モジュール」（動作クロック20MHz，A/D機能内蔵，8KBフラッシュROM内蔵，RS232CドライバIC，5Vレギュレータ実装済）。単品での購入価格は1,700円



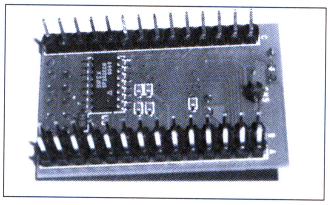
《写真1》「AKI-PIC877モジュール用ベースボード・キット」。4,600円　http://www.akizuki.ne.jp/　秋月電子通商



《写真3》PIC877モジュールの大きさ（一円玉と比較）



《写真4》カッターナイフでピンヘッダを分離させる



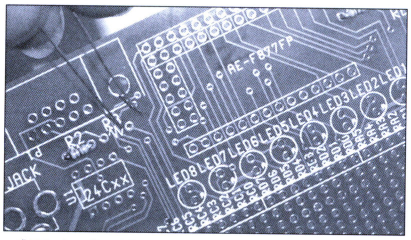
《写真5》モジュールの基板裏にピンヘッダをはんだ付け

組み立て

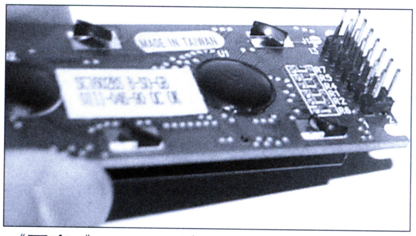
モジュール側の基板には，電源を供給するためのDCジャックが付属しますが，今回は，ピンヘッダから電源を供給するので不要です。基板にキズを付けてから折って取りましょう（**写真3**）。そして，基板にはピンヘッダをはんだ付けします（**写真4，5**）。

次はベースボード側の組み立てです。ベースボード側も部品点数が少ないので，比較的簡単に組み立てができます。背の低い「抵抗」から取り付けましょう（**写真6**）。LEDとEEPROMは極性に注意しましょう。EEPROMは「I2C」という方法でPICと通信します。1Mビットもの大容量で，大量のデータを格納することができます。

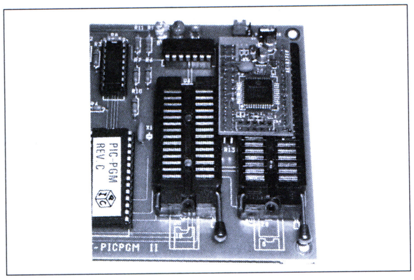
あとは，DCジャック，Dsub9pinコネクタ，タクトスイッチ，可変抵抗，液晶（LCD）モジュールを基板に取り付けたら組み立ては完成です（**写真7**）。



《写真6》部品は背の低いものから順番に取り付ける



《写真7》LCDモジュール（16文字×2行）にピンヘッダをはんだ付け

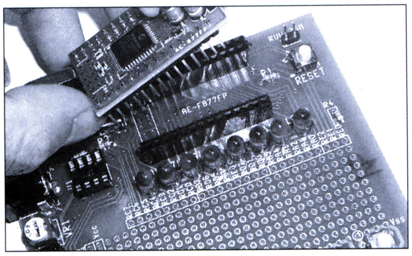


《写真8》「PICプログラマキット」で，AKI-PIC877モジュールのフラッシュROMにプログラムを書き込み

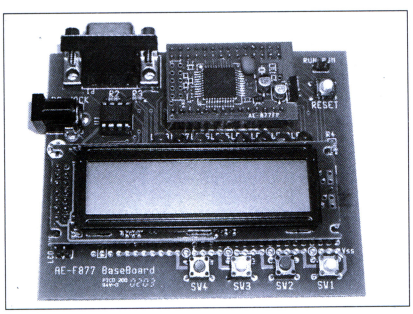
ファームウェアの書き込み

「AKI-PIC877モジュール」に電源を入れても，そのままでは動きません。マイコンに内蔵されている8KBフラッシュROMに一度，ファームウェア（プログラム）を書き込む必要があります。

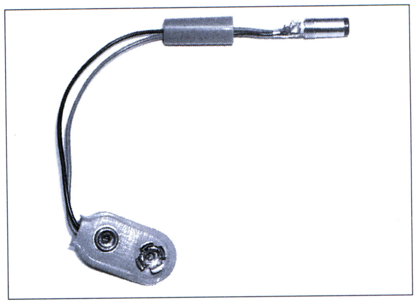
書き込みは秋月電子のPIC専用ライタ「AKI-PIC プログラマーキット」で行います（対応OSはWindows98）。書き込むときは，ROMソケットに片方の列だけを乗せます（**写真8**）。少々変わってますが，これはピンの幅が違うためです。



《写真9》AKI-PIC877モジュールを接続



《写真10》完成した「PIC877ベースボード」

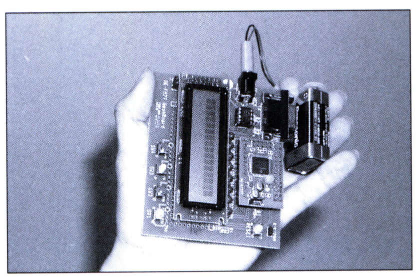


《写真11》DCジャックに乾電池用タップを取り付け。パーツ代は約130円

書き込みが完了後，モジュールのマイコンボードとベースボードを合体させれば完成です（**写真9，10**）。

乾電池で動かす！

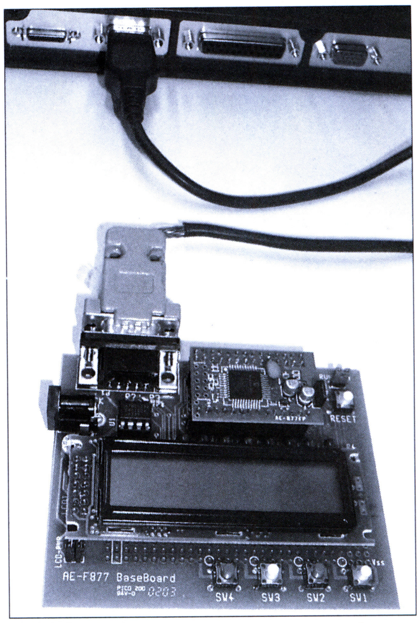
DCジャックは内側（軸側）がVCC，外側がグランドです。電源には，手近にある7～12V出力のACアダプタを流用すればいいだけですが，せっかくなので乾電池で動かしてみたいと思います。方法は簡単で，2.1 ϕ のDCジャックと，9Vの乾電池，そしてタップを用意するだけです（**写真11**）。電池駆動だと持ち運びに便利です。携帯性を生かした新しい利用法が生まれそうです（**写真12**）。



《写真12》乾電池で動かせば，このとおり手のひらサイズに

次回予告

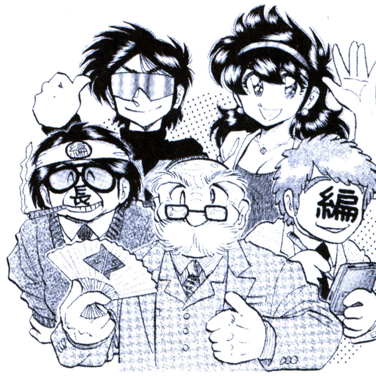
このベースボードは，なんと「PIC BASIC」というBASICインタープリタを動作させることが可能です。BASICコンパイラなら，すでにAKI-H8で採用されていましたが，インタープリタではありません。「PIC BASIC」はインタープリタ型で，しかもH8よりも安いPICで動くわけですから，期待が高まります（**写真13**）。



《写真13》「PIC BASIC」動作時のイメージ。パソコンとベースボードをシリアルケーブルで接続

次回は，ついに「PIC BASIC」のファームウェア書き込みに挑戦します（続きます）。

あくせすROOM



読者参加お待ちします

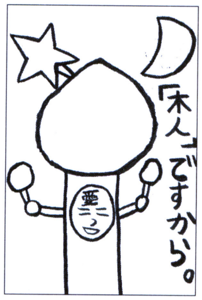
影：花粉症のシーズン再来！　去年までは大丈夫だったけど，今年はどうも目がかゆい……そんなドキドキワクワクの時期だけれども，みんなは大丈夫かな？

つぐ美：明日は我が身と恐怖におののく春のうららかな日……もうそんな時期なのねえ。

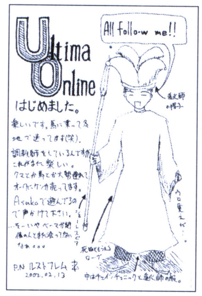
影：花粉症の症状が出ても，私のようにマスクと花粉よけサングラスで完全防備を固めれば，花粉の多い日でも安心だぞ！　さらに鼻水どめの薬を飲んでいれば……眠い……ねむ……グー。

編：教室の窓際に座っている生徒さんも，春の日差しを浴びていると眠くてしかたないですよね。眠気覚ましに効くアイデアがあったらぜひお寄せください。

つぐ美：それでは今月も，みなさんの健康と快適な毎日を願ってやまないあくせすROOMのはじまりです。



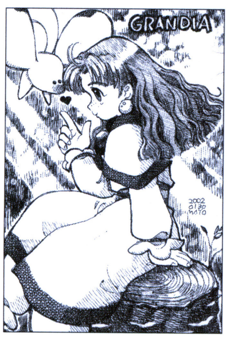
▲愛知県名古屋市・最馬場可
編：どんどんイラストの不条理さに磨きをかけている彼の今後が想像つきません。
つぐ美：仮装大賞らしいけれど，そもそも，なんの仮装なのかしら。



▲東京都練馬区・ルストフレム
つぐ美：ベーマガ読者はAsukaで遊んでいる人が多いのよね。『ファンタシースターオンライン』とちがって，待ち合わせ時間を決めなくても，銀行前で偶然出会ったり，留守宅に遊びに行ってノートに伝言を残していったりできるのが便利なのかしら。



◀天野真
つぐ美：かっこいい女の子ってステキ。自転車で並んで走りたい気分ね。
編：ベーマガ編集部では，いつでも年賀状を募集していますよ。それとは別に，ハガキにはちゃんと住所も書いてほしいです。この時期はとくに住所移転が多いので。



▲山形県米沢市・まよさたーん
つぐ美：女の子の頭にこのペット？が乗っかってるんだけど，食事の時はちゃんと2人で交互に食べてるのが，すごくカワイイの。
編：スクロールが荒くて目が回りましたが，いろいろ細かいところまで作り込まれた名作でしたね。

世界のカスタマ野郎

世界のカスタマ野郎



塩とんこつキムチみそネギこってり昔ながらの醤油ラーメン

動機：「味が薄いなあ」とか思っていると，じつはスープを入れ忘れていることの多い今日このごろのカップラーメン生活。お湯を沸かすのが面倒なときは乾麺のまま食べるので，よけいタレなどが手元に残ってしまいます。捨てるのももったいないので，ひとつのラーメンに全部放り込んでみました。

結果：なんかドロドロの得体の知れない食べ物の完成です。舌先がピリピリする味です。味も大きさも歯ごたえも違うメンマが怪しいハーモニーを奏でています。とにかくニオイがとんでもなくクサい！

◀愛知県名古屋市・改造人間1号

影：身の周りのものを改造して楽しもう！　さっそくプログラムのヘンテコ改造自慢から，ハードウエアの便利な改造まで，いろいろ集まってきているぞ。今月は軽いジャブということで，カップ麺の改造……というか，材料のアレンジを紹介。

市販のカップラーメンに，ゆで卵や野菜を盛り込むのはよく聞くが，今回の「とにかく混ぜてみる」というのは，大失敗のようだ。写真を見てみると……うぷ。こりゃラーメンに見えないよ。チャレンジ精神は評価するが，もっと創意工夫もこらしてほしい。

このコーナーに限り，投稿はハガキサイズにこだわらないので，引き続き，みなさんのレポートを待っている。

あくせすROOMの参加方法

あくせすROOMでは，みなさんの投稿をお待ちしています。愛読者カードや，官製ハガキ，または同サイズの丈夫な紙などを封書でまとめるなどして，お送りください。　鉛筆や色鉛筆はきれいに印刷されません。せっかくペンで描いてあっても，鉛筆の下書きが残っているものは，やはり載りにくくなりますので，インクや水性ボールペンなど，はっきりと書ける道具を使いましょう。文字だけの投稿であれば，ワープロを使ってもかまいませんよ（そのまま掲載できます）。

●応募先

〒141-8715 東京都品川区東五反田1-11-15
電波新聞社　マイコンBASICマガジン編集部
あくせすROOM係

《募集物の一例》

◎うろ覚え選手権

●5月号の募集：街角で耳にしても，なんのことやらサッパリわからない「けさらんぱさらん」という謎のものを，資料を見ないで想像力だけをたよりに，イラストにしてみてください。3月15日必着です。

●6月号の募集：道の終わりにはなにがあるのか？海を行くところまで行き着くとなにがあるのか？誰もたどりついたことのない「世界の果て」を想像力を駆使して描いてみましょう。4月10日必着です。

★世界のカスタム野郎

愛用の道具をいろいろカスタマイズ（自己改造）して使っている人の自慢話を募集します。冷蔵庫にパーツを組み込んでDOS／Vパソコンを作る！　ゲームコントローラに新しく機能をつける！　ちょっとした工夫で，実用性がアップしたり，カッコ良くなったという実例を，ハガキ1枚をメドに紹介してください。

★デジカメ激写ボーイズ

デジカメを持って街に出よう！　デジカメやデジピで撮影したユニークなお笑い画像を送ってください。タイトルはアレですが，女性のかたの投稿も歓迎です。

あくせすROOMは，みなさんのパワーで作るページです。このほかにも，おもしろいネタがあったら，文字でもイラストでも，思いつくままに表現してみてください。企画として長くつづくポイントは，ほかの人がちょっとしたアレンジで相乗りできること。みんなで楽しく参加できそうなネタを常時募集しています。

4月号お楽しみプレゼント

①リバーシブル型携帯用枕「モバイルプロ」…持ち運び時はバッグとしての機能付き／オレンジ・ブルー・グレー各色1名，計3名（提供:ロフテー）

②アナログRGBセレクター「SELECTY15」…3台までのパソコンで1台のディスプレイを共有可能／1名（提供：電波新聞社）



▲モバイルプロ

●応募方法：本誌とじ込み愛読者ハガキ（またはインターネット読者ハガキ）にプレゼント記入欄があります。一つだけ選んでお送りください。

●締め切り：2002年4月6日必着

《2002年2月号の当選者》①-A埼玉県大里郡・真下智行さん　①-B静岡県清水市・池田浩之さん　①-C岩手県岩手郡・伊藤祐司さん　②群馬県前橋市・高埜博典さん他4名　③岐阜県各務原市・寺田吉希さん他4名

Falcom® NEWS EXPRESS

日本ファルコム株式会社
〒190-0012 東京都立川市曙町1-14-13
TEL 042-527-6501
FAX 042-528-2714

スタッフ募集中
採用担当BFM係
http://www.falcom.co.jp/

緊急速報! 新価格WindowsXP対応「イースⅠ完全版(オマケ付き)」3/28発売!



アクションRPGの定番中の定番ソフト「イース」が，WindowsXP対応，しかもお手ごろ価格の新パッケージで登場するぞ！さらに「ファルキャラトレーディングコースター」他，NEWグッズがオマケにつくぞ！新価格XP対応イースⅠは，ファルコムホームページのオンラインショッピングコーナーでも予約受付中！

ファルコム通販価格：3,800円

- オマケ1 ファルキャラトレーディングコースター
- オマケ2 イースたためるCDスタンド
- オマケ3 イースチビデカ2wayマット



▲オマケのNEWグッズはファルコム通販で購入できるぞ！

もう遊んでみました？Zwei!!体験版・無料ダウンロード！

Zwei!!体験版をダウンロードできるサイトも急増中！

やっぱり実際に遊んでみないとねという方は，ぜひ試してみてね。また，ファルコムホームページでは，体験版のマニュアルにデモムービー，七瀬葵新作イラストの壁紙，スクリーンセーバー，新作CDの試聴版も無料でダウンロードできるので，ぜひご利用ください。

下記サイトでZwei!!体験版をダウンロードできます。

●デジキューブオンライン
http://flets.digicube.co.jp/
●ベクター
http://www.vector.co.jp/soft/win95/art/se228483.html
●オンラインプレイヤー
http://www.onlineplayer.jp/
●4Gamer Net
http://www.4gamer.net/news/history/2002.01/20020111200446detail.html
●enban. net
http://enban.net/cgi-bin/view.cgi?file=game/taikenban-d/index.html



▲新価格・新パッケージ版も絶賛発売中！

「ロードモナークスペシャル～オリジナル&Pro～」好評発売中！

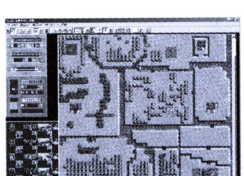
久々のダウンロード版の新作（？）が遂に登場！

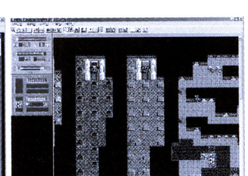
今回は何と「ロードモナークオリジナル」と，「ロードモナークPro」の２つのタイトルが１度に買えて，価格もびっくりの1980円！

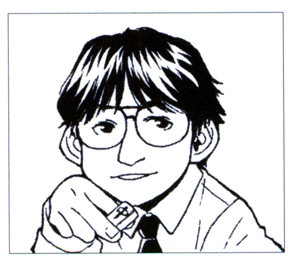
マップ数は2つのタイトルを合わせて全部で104面とたっぷり遊べます。パッケージ版は，今では店頭ではほとんど入手ができなくなっているので，ロードモナークFirstや，オンライン版だけでは物足りない，もっともっとロードモナークを遊びたいという方は，ぜひダウンロード販売をご利用ください！

三つのサイトで発売！

ベクター
http://www.vector.co.jp/
BIG LOBE
http://softplaza.biglobe.ne.jp/shop/falcom/
パピレス（3/14予定）
http://www2.papy.co.jp/f_pc/index.htm

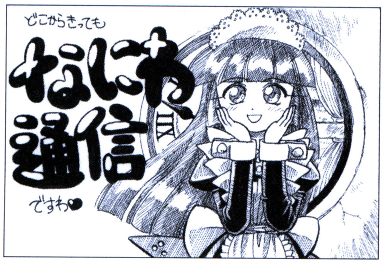






Zwei!!に続く新作ゲームの開発中バージョンをちょっとだけプレイしてみましたが，もうだいぶ完成に近づいていて，かなりハマってしまいました。この号が出ている頃には，タイトルも発表されているかも？

いろんな意味でアツクなれるゲームですので，乞うご期待！



XRGB-2plus発売中！

なにわ通信

PRESENTED BY マイコンソフト

http://www.micomsoft.co.jp

ADDRESS● 〒530-0005大阪市北区中之島3-2-4
朝日新聞ビル6F電波新聞社大阪本社内
マイコンソフト株式会社 なにわ通信係

タイトル冠文：大阪府・御堂筋太郎
タイトル・イラスト：埼玉県・野沢プロト
本文：暁

皆さんこんにちは！もうすぐ春ですね。

進学，卒業と忙しい時期ですが，みなさんいかがお過ごしでしょうか。

休日はお店へ行こう！

マイコンソフトの製品は，他誌に広告を出している訳ではないのですが，なぜか多くの方が製品の発売を知っているようでした。

おそらく，なにわ通信が情報の発信源になっているのだと思います(ちょっぴり嬉しいです)。マイクロソフトさんのXBOXが発売されたこともあって，XRGB-2plusにコンポーネント・ケーブルで接続できるゲーム機が1機種増えます。

これを期に皆さんも「XRGB-2plus」の購入を検討してみてはいかがでしょうか？

やっぱり機械も新鮮な方が良いと思いますよ(笑)。

「XRGB-2plus」はゲームショップや家電店の棚で皆さんのお越しをお待ちしております。

こんな使い方も

XRGB-2plusに内蔵されている鮮明度調整機能(シャープネス調整)は，ちょっと変わった方式で映像を処理しています。このため，元々ぼかしが利いているようなゲーム機(任天堂さんのニンテンドー64やスーパーファミコン)の映像も調整によっては本来の映像よりも鮮明に見ることができます。お持ちの方は，試しにXRGB-2plusの「SHARP」機能を調整してみてください。昔のゲーム機が，意外と鮮明に楽しめるかもしれませんよ。

フライヤー募集

前号でもお知らせしましたが，なにわ通信では4回目のフライヤーを募集しています。

表現したいことを紙に描いていただき，マイコンソフトまでお送り下さい。

希望される読者の方に配布させていただきます。

締め切りは4月末です。参加者，及び配布希望者を募集しております。

ありがたい，おハガキのコーナー

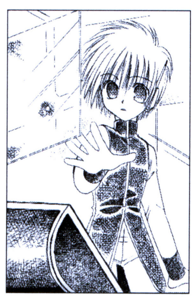
なにわ通信では，イラストや4コマ漫画，冠文などを募集しております。ドシドシご応募ください！



東京都・天野真
ずいぶん前に描かれたイラストなのだそうです。机を掃除していて出てきたそうです。机の中にイラストがザクザクなんてすごい。



千葉県・鈴木有人
2月号の続きです。あの坊やがこんなに立派な剣士に！ちなみに20年経過しているとのこと。結局，食虫植物さんは食べ逃したみたい。



京都府・陽東あひる
ガラスの世界でしょうか？キャラクターの目がうつろですねぇ。この子は魔法使いなのでしょうか。ちょっぴり神秘的です。



山形県・ひねくまよ
くまマンと，わるくまマンが主役みたいで，女の子はオマケみたいですね(笑)。山形には，まだ雪が積もっていますか？



愛知県・邪道坊や
激戦の中，舵を握る男はカッコイイですよね！宇宙戦艦ヤマトをもう一度見たくなってしまいました。



宮城県・吏久
うをっ！猫ミミですね！むむむ〜。本当に猫ミミの子が居たら，魚で誘惑して捕まえてしまうかもしれません。

最後の熊恋

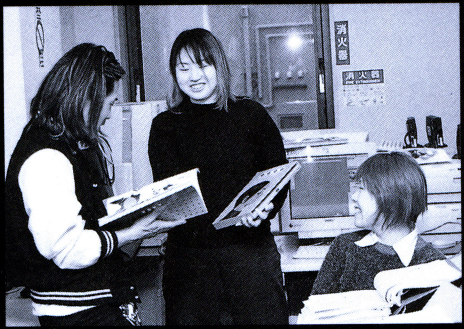


大阪府・ARAI

ここならゲーム業界で働ける

在学中に現役ゲームデザイナーが、就職できるまでの技術を完全指導!!

●声優、漫画家、イラストレーターになりたい人の学校！ ●東京アニメーター学院（姉妹校）大人気！願書受付中 ☎03(5276)1511



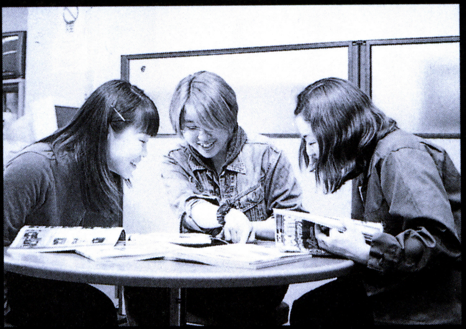
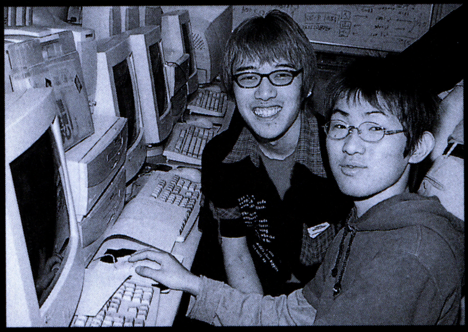



14年4月生願書受付中

無料体験実習日程・学院見学!

まだ間に合う！4月入学生！体験実習日は3月23日（土）午後1時。学院見学・入学相談は毎日実施。お申込みは電話、FAX、Eメールで



授業内容
各科共に年間カリキュラムが組まれ、それを前後期で実施。2年生はチームを組んで現場と同じにゲームを制作。ここで就職できる実力を習得している。

講師
本学院の講師はすべて現役のゲームデザイナーが指導。業界に精通しているので、就職情報や技術面でも生徒の信頼が厚い。現場への研修も斡旋する。

マシン環境
ゲーム業界はソフト、ハードとも常に変化している。学院では毎年、最新設備を整え新入生を迎えている。現場で使われている機材で学習するのがベスト。

就職活動
現在、3月卒業にむけて就職活動もピークをむかえている。生徒が希望する就職を達成させるために、進路指導部と担任の先生が、個別にきめ細かな指導をしている。幸い求人数も多く、生徒の技術力によって、それなりの就職内定を獲得している。（14年2月現在）

学費（年間費用）
創立以来、ゲーム関連学校の中では、常に良心的で適正低廉な学費を設定。14年度入学生は84万円、2年目は66万円。入学後の徴収はなく、すべてを含む。

教育ローン・学生寮
①本学院が三井住友銀行と提携している教育ローンだと、毎月6276円で学べる。80万円を14年間返済②環境と設備を整えた学生寮（男女各90名）あり。

新聞奨学制度
朝日、毎日、読売新聞等を配達することで、学費が全額免除される。また、住む場所と給与12万円が支給される。手続きなど、詳しくは入学案内書に記載。

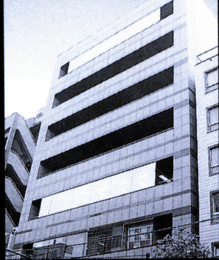
体験実習・学院見学
体験実習日は3月23日（土曜日）午後1時開催。参加希望者は電話でお申込み下さい。学院見学は毎日実施しています。電話で予約し、来校して下さい。

■就職抜群!学生寮完備。朝日・毎日・読売新聞等奨学金制度（学費全額免除・給与月額12万円支給）三井住友銀行教育ローン有。学院見学随時可⇨日時をTELにて要予約

14年4月生願書受付中

ゲームデザイナー学部
全日制2年・定員300名

ゲームデザイン科
プログラム科
グラフィック科
キャラクターデザイン科
デジタルサウンド科



文部省認可CGARTS協会認定CG教育校／社団法人・日本漫画家協会賛助会員／社団法人・日本映画テレビ技術協会会員

日本ゲームソフト学院

〒101-0061 東京都千代田区三崎町3-4-9 ☎03-3261-8211

入学案内書 はがき（住所・氏名・年令・電話番号明記）か、電話で上記まで（無料）

FAX 0120-194-750 http://www.tag.co.jp/ E-mail:ngs@tag.co.jp

◆ 定期購読●直接購読のしおり ◆

弊社刊行の雑誌は，ご自宅直送方式の年間定期購読をお勧め申し上げます。スピーディーで確実，そしてお得な定期購読をぜひご利用下さいませ。

また　ご希望の雑誌・書籍が，ご近所の書店にない場合は右の振替用紙にて直接ご注文下さい。

■ 年間定期購読料（消費税込料金となっております。）

電　気　店	￥16,000	マイコンBASICマガジン	￥6,120
電波新聞縮刷版（CD-ROM版）		￥96，000	

■ 年間購読の特典

①特別号も通常の定価でご購読いただけます。
②電気店は13冊分の購読ができます。

◆確実な振替口座を利用してご送金下さい◆

送金手数料が不要で，確実な郵便振替でご送金をお願い申しあげます。手続きは簡単です。右の郵便振替用紙に下記の必要事項を記入し，郵便局窓口に必要金額と共にご提出いただくだけですみます。

①購読申込書に必要事項を記入ください。なお１年間，或は２年間購読かをはっきりご記入ください。
②払込通知票，払込票に送金合計金額と住所，氏名をご記入ください。払込通知票と払込票は絶対に切りはなさないようにしてください。なお，必ずフリガナをご記入下さい。
③弊社への御意見・御希望がございましたら，通信欄へご記入下さい。
④すべて記入が終了しましたら，案内文のみ切り離して郵便局窓口に金額を添えて提出してください。

∴振替によるご送金は，着本までに約２週間ほどかかりますのでお含みおきください。
お急ぎの場合は，現金書留に，右購読申込書を同封してご送金ください。

この受領証は、郵便局で機械処理をした場合は郵便振替の払込みの証拠となるものですから大切に保存してください。

ご注意

この払込書は、機械で処理しますので、口座番号及び金額を記入する際は、枠内に丁寧に記入してください。

また、下部の欄（表面及び裏面）を汚したり、本票を折り曲げたりしないでください。

（郵政事業庁）

払込取扱票

02 東京

通常払込料金 加入者負担

口座番号（右詰めにご記入ください）

0 0 1 5 0 ― 3 ― 5 1 9 6 1

金額 千 百 十 万 千 百 十 円
※

加入者名 株式会社 電波新聞社

料金

特殊取扱

通信欄

※定期購読誌名	購読期間	送金額（送料当社負担）
※新規 継続	年 月号より 年間	円

御希望誌・ソフト・グッズ名	冊数	送金額（送料共）
		円
		円

フリガナ
送付先住所 □□□-□□□□

産業分類 所属又は職種

フリガナ

電話 TEL 年令 才 性別 男・女

払込人住所氏名（郵便番号 ）
※
（電話番号 ― ― ）

受付局日附印

裏面の注意事項をお読みください。郵政事業庁（私製承認 東 第1005号）

各票の※印欄は、払込人において記載してください。

切り取らないで郵便局にお出しください。

記載事項を訂正した場合は、その箇所に訂正印を押してください。

払込票兼受領証

口座番号 0 0 1 5 0 ― 3 通常払込料金加入者負担

右詰めにご記入ください 5 1 9 6 1

加入者名 株式会社 電波新聞社

金額 千 百 十 万 千 百 十 円
※

払込人住所氏名
※

料金（消費税込み） 円

受付局日附印

特殊取扱

切り取って郵便局へお出し下さい。

電波新聞社出版物一覧

（消費税込料金となっております）

2002. 3 現在

最新エレクトロニクス用語事典	5,300円	〒380円	F-BASIC V6.3プログラミングのすべて	2,835円	〒310円
半導体・ICのすべて	2,520円	〒310円	まんがぱそてんシリーズ 第1巻(基礎編)/第2巻(応用編)/第3巻(ネットワーク編)	各 1,260円	各 〒310円
ビデオ技術のすべて	2,100円	〒310円			
TVのすべて改訂版	1,988円	〒310円	パソコン認定試験合格対策試験問題集3・4級	2,100円	〒310円
DVD&ホームシアター	1,680円	〒310円	シスアド受験スーパー記憶術	2,900円	〒340円
新デジタル映像技術のすべて	2,100円	〒310円	基本情報受験スーパー記憶術	2,835円	〒340円
センサーのすべて	2,000円	〒310円	初級シスアド受験マニュアル	3,780円	〒450円
改訂版DVD&DVC入門基本18章	1,529円	〒310円	基本情報受験マニュアル	3,780円	〒450円
増補改訂版デジタル放送がわかる本	1,600円	〒310円	デジタル1種〈技術科目〉最新テキスト	2,205円	〒340円
増補改訂版・新ビデオ技術ハンドブック	2,500円	〒310円	デジタル3種受験マニュアル	2,940円	〒340円
新・まんが電気学入門	1,500円	〒310円	アナログ・デジタル全資格〈法規科目〉	1,995円	〒340円
新電気学入門講座	2,050円	〒310円	特殊無線技士受験マニュアル・2002年版	2,940円	〒340円
新オシロ・ロジアナ入門講座	2,243円	〒310円	オールアバウト カプコン対戦格闘ゲーム1987~2000	2,100円	〒450円
電子計測器ガイドブック	7,500円	〒450円	オールアバウト SNK対戦格闘ゲーム 1991~2000	2,310円	〒450円
よくわかるプラズマテレビ(PDP)	1,680円	〒310円	電子パーツ物知り百科	1,890円	〒310円

※上記以外にも新聞・出版物がございますので最寄りの本社、支局へお問い合わせ下さい。

東京本社 03-3445-8201(ダイヤルイン)
大阪本社 06-6203-3361
西部本社 092-431-7411
札幌支局 011-641-5591
仙台支局 022-227-7211
新潟支局 025-245-2526
関東総局 027-326-3206
松本支局 0263-36-2266
名古屋支局 052-261-4541
金沢支局 076-263-8661
京都支局 075-221-8021
広島支局 082-228-5581
高松支局 087-861-3111
鹿児島支局 099-226-3630

キ リ ト リ 線

月刊マイコンBASICマガジン④ 愛読者カード

★OFコーナー宛 ♥あくせすROOM宛 ■質問・感想（本誌への掲載／可・不可）

ペンネーム：

■アンケート回答記入欄 ■

※黒の筆記用具でわかりやすく，はっきりとご記入ください

Q1	□	Q2	□	〔メーカー名 〕				
Q3	□	Q4	□	Q5	□			
Q6	□	□	□	□	□	□	□	□
Q7	□	□	□	□	□	〔商品名 〕		
Q8	□	□	□	□	□	□	□	□
Q9	□	□	□	□	□	□	□	□
Q10	□	Q11	□	〔④の人はソフト名を記入 〕				
Q12	〔 〕	Q13	〔 〕					
Q14	□	□	□	Q15	□	□	□	

キ リ ト リ 線

■アンケートにご協力ください。

※左の愛読者カード・記入欄に，以下の質問に該当する番号をご記入ください。複数回答の質問で，回答欄をすべて埋める必要はありません。

Q1　メインで使っているパソコンは？
①デスクトップ ②ノート ③持っていない

Q2　パソコンの種類は？（Q1で③以外の方）
①自作 ②ショップブランド ③メーカー製
※枠内にメーカー名も記入してください

Q3　メインで使っているOSは？（Q1で③以外の方）
①Windows 95 ②Windows 98 ③Windows Me ④Windows 2000 ⑤Windows NT ⑥Windows XP ⑦Macintosh ⑧UNIX ⑨Linux ⑩その他

Q4　パソコン歴は？
①1年未満 ②1～2年 ③3～5年 ④6～10年 ⑤11年以上 ⑥なし

Q5　インターネット歴は？
①1年未満 ②1～2年 ③3～5年 ④6～10年 ⑤11年以上 ⑥なし

Q6　パソコンの主な使用目的は？（複数回答可）
①プログラミング ②ゲーム ③インターネット ④メール ⑤ミュージック ⑥学習 ⑦CG ⑧その他

Q7　これから買いたいと思うコンピュータ機器は？（複数回答可）
①パソコン本体 ②プリンタ ③スキャナ ④デジタルカメラ ⑤その他
※わかれば，商品名も記入してください

Q8　持っているゲーム機は？（複数回答可）
①プレイステーション（PSone含） ②プレイステーション2 ③ドリームキャスト ④ニンテンドウ64 ⑤ゲームキューブ ⑥Xbox ⑦ゲームボーイ ⑧ゲームボーイカラー ⑨ゲームボーイアドバンス ⑩ワンダースワン ⑪ワンダースワンカラー ⑫その他

Q9　これから買いたいと思うゲーム機は？（複数回答可）
①プレイステーション（PSone含） ②プレイステーション2 ③ドリームキャスト ④ニンテンドウ64 ⑤ゲームキューブ ⑥Xbox ⑦ゲームボーイカラー ⑧ワンダースワンカラー ⑨ゲームボーイアドバンス

※裏面へつづく→

■メモ欄…プログラム担当者・編集部へのご意見・ご感想など

キリトリ線

Q10　携帯・PHSを持っていますか？
①両方持っている ②携帯 ③PHS ④購入予定 ⑤予定なし

Q11　ワープロ・ソフトは何をお使いですか？
①Word ②一太郎 ③メモ帳 ④その他

Q12　使用中のプログラミング言語は？
※枠内に一つご記入ください

Q13　これから勉強したいプログラミング言語は？
※枠内に一つご記入ください

Q14　今月号の本誌で面白かった記事を三つ，下の表からお答えください

Q15　今月号の本誌でつまらなかった記事を三つ，下の表からお答えください

ご協力ありがとうございました。

■今月号の記事一覧

マイコンBASIC Magazine　プログラム投稿応募カード

言語名（音源名）					
プログラム名（曲名）					
氏　名		ペンネーム		年齢	歳
住　所	〒				
TEL	※連絡可能な時間帯		応募年月日	年　月　日	

郵便はがき

50円切手をお貼り下さい

141-8715

（受取人）

東京都品川区東五反田1－11－15

電波新聞社

月刊 マイコンBASICマガジン 編集部

4月号 行

		年齢	性別
(フリガナ)			
ご氏名		歳	男・女
□□□-□□□□	電話		
ご住所	都・道 府・県		

職業：小学生／中学生／高校生／高専生／短大生／大学生／専門学校生／予備校生／アルバイト／公務員／会社員／その他（　　　）

本誌を 毎月・時々・初めて 読む　何人で読みますか（　）人

自分のホームページを　持っている・持つ予定・持っていない・予定なし

本誌のほかによく読む雑誌は？［　　　］

	希望番号	品名
ベーマガお楽しみプレゼント④ 応募欄（126ページ掲載）	番	

キリトリ線

パソコンRPG入門最適！

PCゲームの定番王「イースⅠ完全版」オマケ付
新価格 3,800円！（ファルコム通販価格）



新価格WindowsXP対応

3/28発売！

イースⅠ完全版

初回限定
オマケ付

DVD-ROM版／CD-ROM版 ¥3,800（ファルコム通販価格）

※新価格XP対応「イースⅠ完全版」は、「イースⅠ&Ⅱ完全版」の「イースⅠ」と同じものです。ゲーム内容に変更はありません。

スピーディーに展開するドラマティックな物語。独自の世界観を構築するグラフィックと音楽。他に類をみない抜群の操作性と爽快感。RPGブームの先駆けであり、全てのストーリー型RPGの基礎となったゲーム、それが「イース」です。イージーからハードまで難易度が選択できるから、RPG初心者でも必ずエンディングまでプレイできます。多くのユーザーが魅了され続け、いまだ語り継がれる"伝説"を、ぜひあなたも"経験"してください。

【オマケ1】ファルキャラトレーディングコースター
トレーディングカードのように集めて楽しい新感覚超極薄コースター（90×90mm/厚さ0.5mm）。紙製ではないので汚れてもサッと一拭きで繰り返し使えます。裏面は貼り跡の残らない特殊粘着加工。フィルムをはがせばデスクやテーブルにピタッと貼りつき安定感バツグン。カードのように薄いので額に入れて飾るなどお部屋のインテリアにも最適。（別売:1パック3枚入480円）

【オマケ2】イースたためるCDスタンド
イースエンブレムがお洒落な最大9枚収納できる折りたたみ式CDスタンド。蓋をたたむと普通のCDプラケースと同じサイズになり、使わないときも邪魔にならないスグレもの。（別売:1,200円）

【オマケ3】イースチビデカ2wayマット
チビマウスマットにも、デカコースターにもなる超便利マット（120×120mm）。クッション付で滑らない安心の使い心地。この便利さ使った人しかわからないかも。（別売:480円）

※オマケ1〜3は別売りもあります。ご購入はファルコム通販をご利用ください。



http://www.falcom.co.jp/

七瀬葵描き下ろし新イラストNEWパッケージで登場！

PCゲーム売上ランキング1位！

週刊アスキー「今週の売れ筋ソフト」（2001年12/17〜12/30集計）、CD-ROM Fan「売れ筋ランキング」（2001年12月集計）
TECH Win「SALES RANKING」（2001年12/1〜12/31集計）、電撃王「PCゲーム売り上げチャートBEST10」（2001年12/1〜12/31集計）



Windows 95/98/Me/2000/XP
DVD-ROM版 CD-ROM版

新価格版 好評発売中

攻略ガイドも発売中。完全制覇を目指せ！

■お求めは取扱い店またはカンタン便利なファルコム通販（falcom.comファルコムドットコム）もご利用下さい。■社員募集（新卒・既卒・中途）！詳しくはホームページ「会社案内・社員募集」をご覧ください。

Falcom®
NIHON FALCOM CORPORATION
〒190-0012 東京都立川市曙町1-14-13
☎042(527)6501 http://www.falcom.com/



第二十一巻第四号通巻二三八号 昭和五十七年十二月六日第三種郵便物認可 二〇〇二年三月二十五日印刷 二〇〇二年四月一日発行
電波新聞社 〒141-8715 東京都品川区東五反田一−十一−十五 電話〇三(三四四五)六一一一

定価550円 本体524円 雑誌18361-04 T1118361040555

2002

わかる!動かせる!
プログラムが組める雑誌

マイコンBASIC Magazine

4

新連載! ユニバーサルデザイン導入で訪問客を増やす!

一歩先ゆくホームページ作り

http://www.basicmagazine.net/

電波新聞社